CS 355: Applications of ZK

Brian Gu, 0xPARC Foundation

INTRO

Brian and 0xPARC

I’m Brian

- Been working in the ZK space since 2019 (learned about zkSNARKs at a workshop at Stanford!)
- Co-founder of 0xPARC
- Built Dark Forest, the first non-cryptocurrency application of zkSNARKs

0xPARC

- A foundation focused on bringing “programmable cryptography” technologies like ZK, FHE, MPC, … from theory to practice.
- Lots of companies and teams in the blockchain + applied ZK world came out of0xPARC (and ETHUni) programs and grants!

Agenda

- Preamble: zkSNARKs are “programmable” cryptography
- Part 1: How to use zkSNARKs in apps
  - Group / ring signatures
  - Anonymous polling / private cryptocurrency
  - Dark Forest
  - Information marketplaces (likely to skip)
- Part 2: Three categories of zkSNARK applications
  - Adding hidden information to decentralized systems
  - Speeding up blockchains
  - Language of Truth

Agenda

- **Preamble: zkSNARKs are “programmable” cryptography**
- Part 1: How to use zkSNARKs in apps
  - Group / ring signatures
  - Anonymous polling / private cryptocurrency
  - Dark Forest
  - Information marketplaces
- Part 2: Three categories of zkSNARK applications
  - Adding hidden information to decentralized systems
  - Speeding up blockchains
  - Language of Truth

Preamble: Programmable Cryptography

We like to think of zkSNARKs as “programmable” cryptography

INTRO

First gen crypto → Second gen crypto

(First-generation) cryptography

- Encryption and signatures
- “Security” and “privacy”

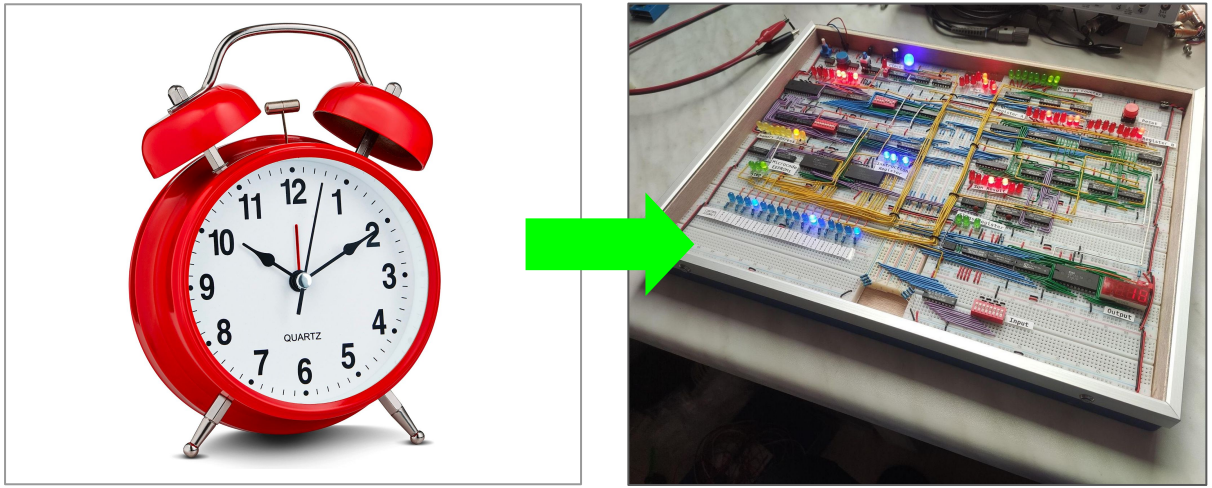
(First-generation) cryptography in the wild

Today, we already use cryptography everywhere without thinking twice – think https in the browser, password managers, E2E encrypted messaging like Whatsapp and Signal, ...

Second-generation cryptography (“programmable cryptography”)

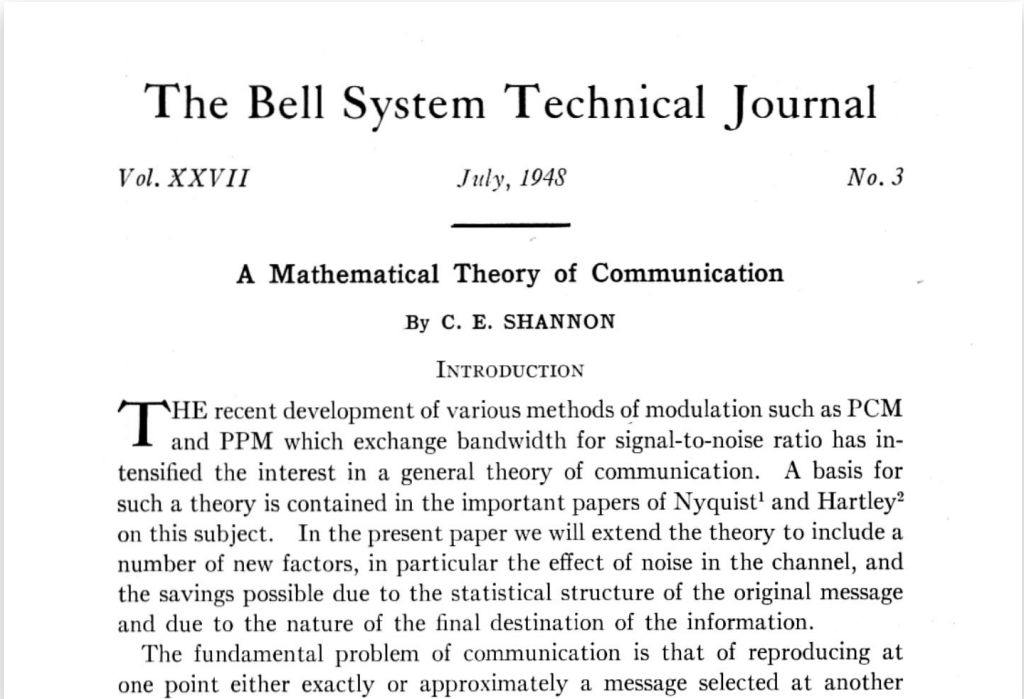
Programmable Cryptography

- Proofs for specific functions → proofs for any function
- Verification of specific claims → verification of any claim
- Special-purpose protocols → general-purpose “cryptography compilers”



Programmable Cryptography

Computation, data, and the operations we perform on them



The Bell System Technical Journal

Vol. XXVII *July, 1948* *No. 3*

A Mathematical Theory of Communication

By C. E. SHANNON

INTRODUCTION

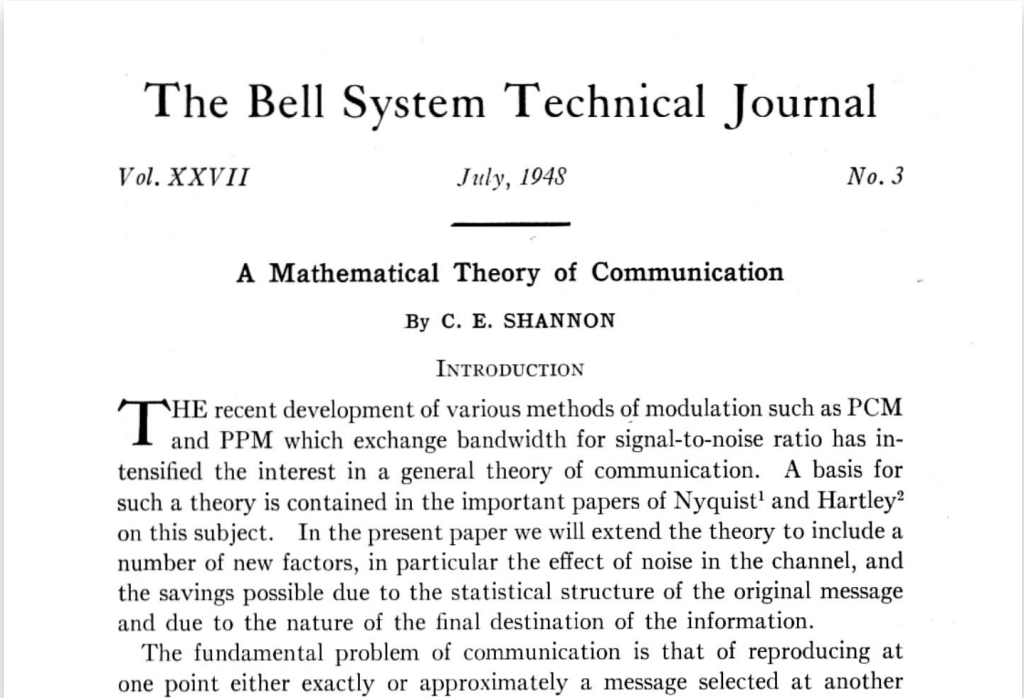
THE recent development of various methods of modulation such as PCM and PPM which exchange bandwidth for signal-to-noise ratio has intensified the interest in a general theory of communication. A basis for such a theory is contained in the important papers of Nyquist[1] and Hartley[2] on this subject. In the present paper we will extend the theory to include a number of new factors, in particular the effect of noise in the channel, and the savings possible due to the statistical structure of the original message and due to the nature of the final destination of the information.

The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another

$$H = -\sum_i p_i \log_2 (p_i)$$

Programmable Cryptography

A tool for building digital systems with powerful new properties; not just a tool for securing systems.



The Bell System Technical Journal

Vol. XXVII *July, 1948* *No. 3*

A Mathematical Theory of Communication

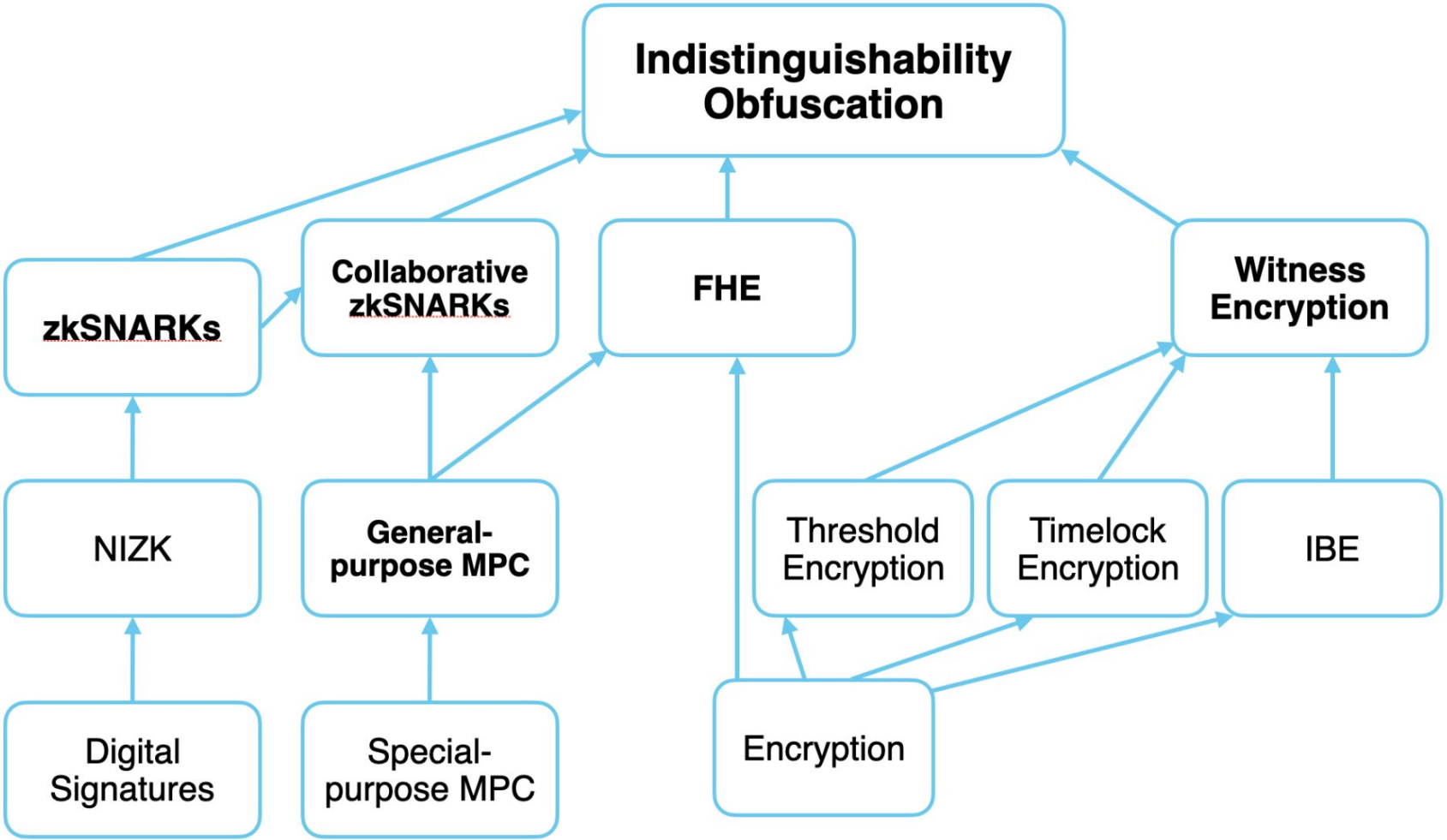
By C. E. SHANNON

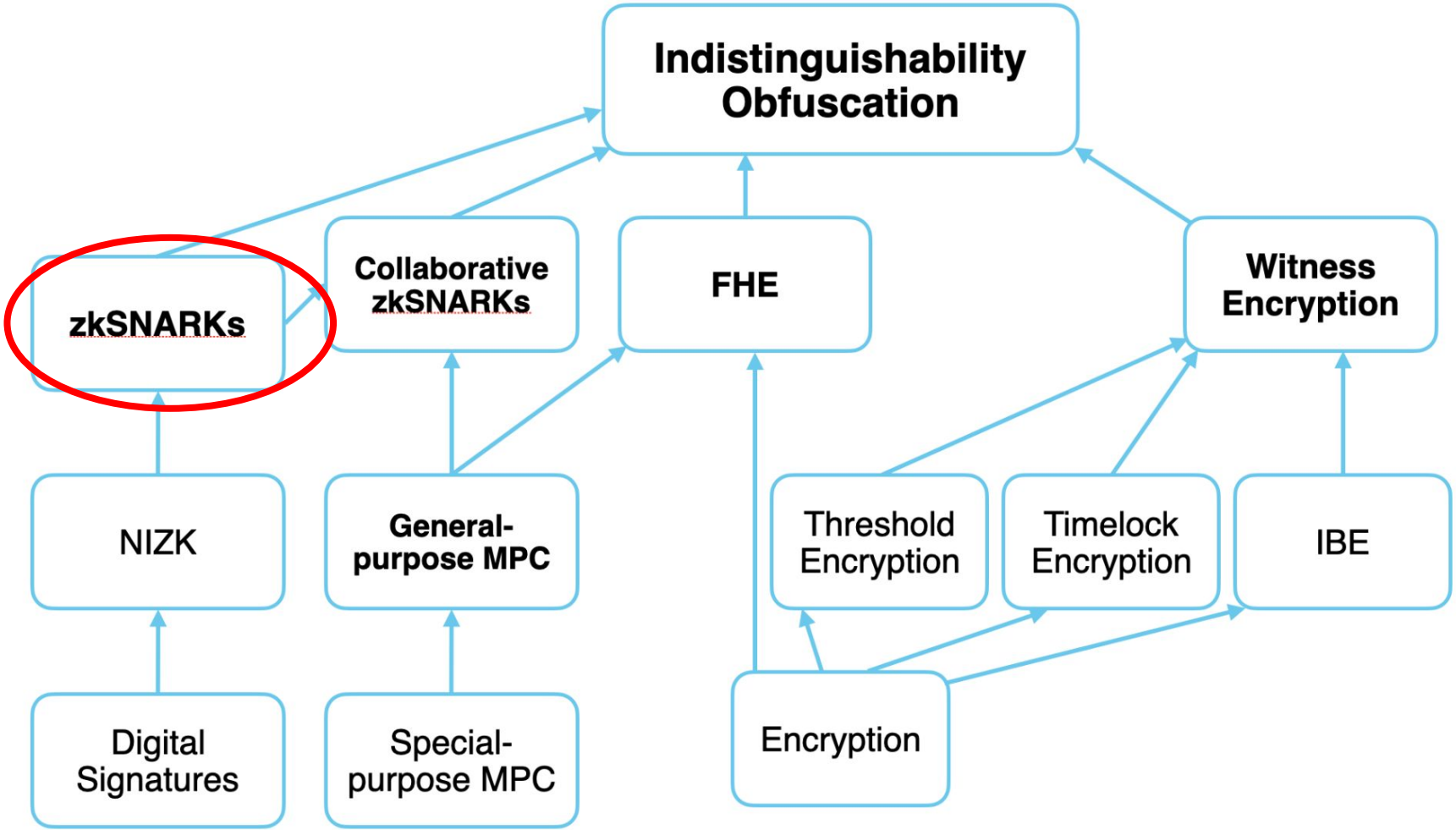
INTRODUCTION

THE recent development of various methods of modulation such as PCM and PPM which exchange bandwidth for signal-to-noise ratio has intensified the interest in a general theory of communication. A basis for such a theory is contained in the important papers of Nyquist[1] and Hartley[2] on this subject. In the present paper we will extend the theory to include a number of new factors, in particular the effect of noise in the channel, and the savings possible due to the statistical structure of the original message and due to the nature of the final destination of the information.

The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another

$$H = -\sum_i p_i \log_2 (p_i)$$


Indistinguishability Obfuscation
zkSNARKs
Collaborative zkSNARKs
FHE
Witness Encryption
NIZK
General-purpose MPC
Threshold Encryption
Timelock Encryption
IBE
Digital Signatures
Special-purpose MPC
Encryption


Indistinguishability Obfuscation
zkSNARKs
Collaborative zkSNARKs
FHE
Witness Encryption
NIZK
General-purpose MPC
Threshold Encryption
Timelock Encryption
IBE
Digital Signatures
Special-purpose MPC
Encryption

Example: zkSNARKs and membership proofs

Let’s look at identity claims!

Example: zkSNARKs and membership proofs

Let’s look at identity claims!

🥳 I know a private key
corresponding to Alice’s public key.

Example: zkSNARKs and membership proofs

Let’s look at identity claims!

😏 I know a private key corresponding to Alice, Bob, OR Charlie’s public keys.

Example: zkSNARKs and membership proofs

Let’s look at identity claims!

😦 I know a private key corresponding to Alice, Bob, OR Charlie’s public keys, and the other two [can/can’t] prove that they did NOT generate this message.

Example: zkSNARKs and membership proofs

Let's look at identity claims!

- 😵 I know a private key corresponding to Alice, Bob, OR Charlie's public key...
- AND I either possess a signed attestation from one of {David, Eve, Fred}, or during the block with header X, I knew the private key corresponding to an account with at least 32ETH...
- OR I possess a biometric that, when run through a neural network, hashes to the fingerprint hash of a non-sanctioned individual.

zkSNARKs turn math problems into programming tasks.

Agenda

- Preamble: zkSNARKs are “programmable” cryptography
- **Part 1: How to use zkSNARKs in apps**
  - Group / ring signatures
  - Anonymous polling / private cryptocurrency
  - Dark Forest
  - Information marketplaces
- Part 2: Three categories of zkSNARK applications
  - Adding hidden information to decentralized systems
  - Speeding up blockchains
  - Language of Truth

Agenda

- Preamble: zkSNARKs are “programmable” cryptography
- Part 1: How to use zkSNARKs in apps
  - **Group / ring signatures**
  - Anonymous polling / private cryptocurrency
  - Dark Forest
  - Information marketplaces
- Part 2: Three categories of zkSNARK applications
  - Adding hidden information to decentralized systems
  - Speeding up blockchains
  - Language of Truth

Context: Group signatures

“I am one of Alice, Bob, or Charlie, and I attest to message M.”

Context: Group signatures

"I am one of Alice, Bob, or Charlie, and I attest to message M."

3.1. Confirmation protocol

We first consider the following instance, which is solved by [BCDvdG87] by using Protocol 1, which uses computationally secure blobs $\mathcal{B}$.

$\mathcal{P}$'s secret	:	c.
public	:	N, x, y, Ω; $x, y \in \mathbb{Z}_N^*, \Omega = \{\alpha, \ldots, \alpha+\beta\} \subset \mathbb{N}$.
prove to $\mathcal{V}$	:	$x^c \equiv y \pmod N \wedge c \in \Omega$.

Instance 1.

If this protocol is iterated k times, $\mathcal{V}$ will be convinced (with probability $1-2^{-k}$) that $c \in \tilde{\Omega} = \{\alpha-\beta, \ldots, \alpha+2\beta\}$, but $\mathcal{V}$ will receive no knowledge other than the fact that $c \in \Omega = \{\alpha, \ldots, \alpha+\beta\}$[†].

Protocol 1. (for Instance 1)

(1) $\mathcal{P}$ *chooses* $r \in \{0, \ldots, \beta\}$. *He computes blobs on* $z_1 \equiv x^r \pmod N$ *and* $z_2 \equiv x^{r-\beta} \pmod N$, *and sends the unordered pair* $\{\mathcal{B}(z_1), \mathcal{B}(z_2)\}$ *to* $\mathcal{V}$.

(2) $\mathcal{V}$ *chooses randomly* $b \in \{0,1\}$ *and sends it to* $\mathcal{P}$.

(3) $\mathcal{P}$ *sends* $\mathcal{V}$ *in case*

b=0: r *and opens both blobs.*

b=1: $\tilde{r}$ *which is* $(c+r)$ *or* $(c+r-\beta)$, *whichever is in the set* Ω, *and opens respectively the blob on* z_1 *or* z_2 (*which is called* $\tilde{z}$).

(4) $\mathcal{V}$ *verifies in case*

b=0: *that* $r \in \{0, \ldots, \beta\}$ *and that the blobs contain* x^r *and* $x^{r-\beta}$ *in some order.*

b=1: *that* $\tilde{r} \in \Omega$, *that one of the blobs contains* $\tilde{z}$ *and that* $\tilde{z}$ *satisfies* $x^{\tilde{r}} \equiv \tilde{z} y$.

560 R.L. Rivest, A. Shamir, and Y. Tauman

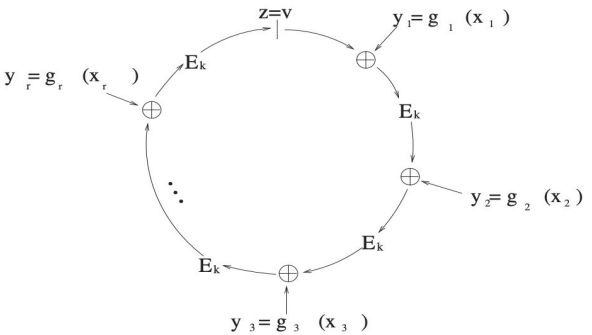


Fig. 2. Ring signatures

2. Pick a random glue value: Second, the signer picks an initialization (or "glue") value v uniformly at random from $\{0,1\}^b$.

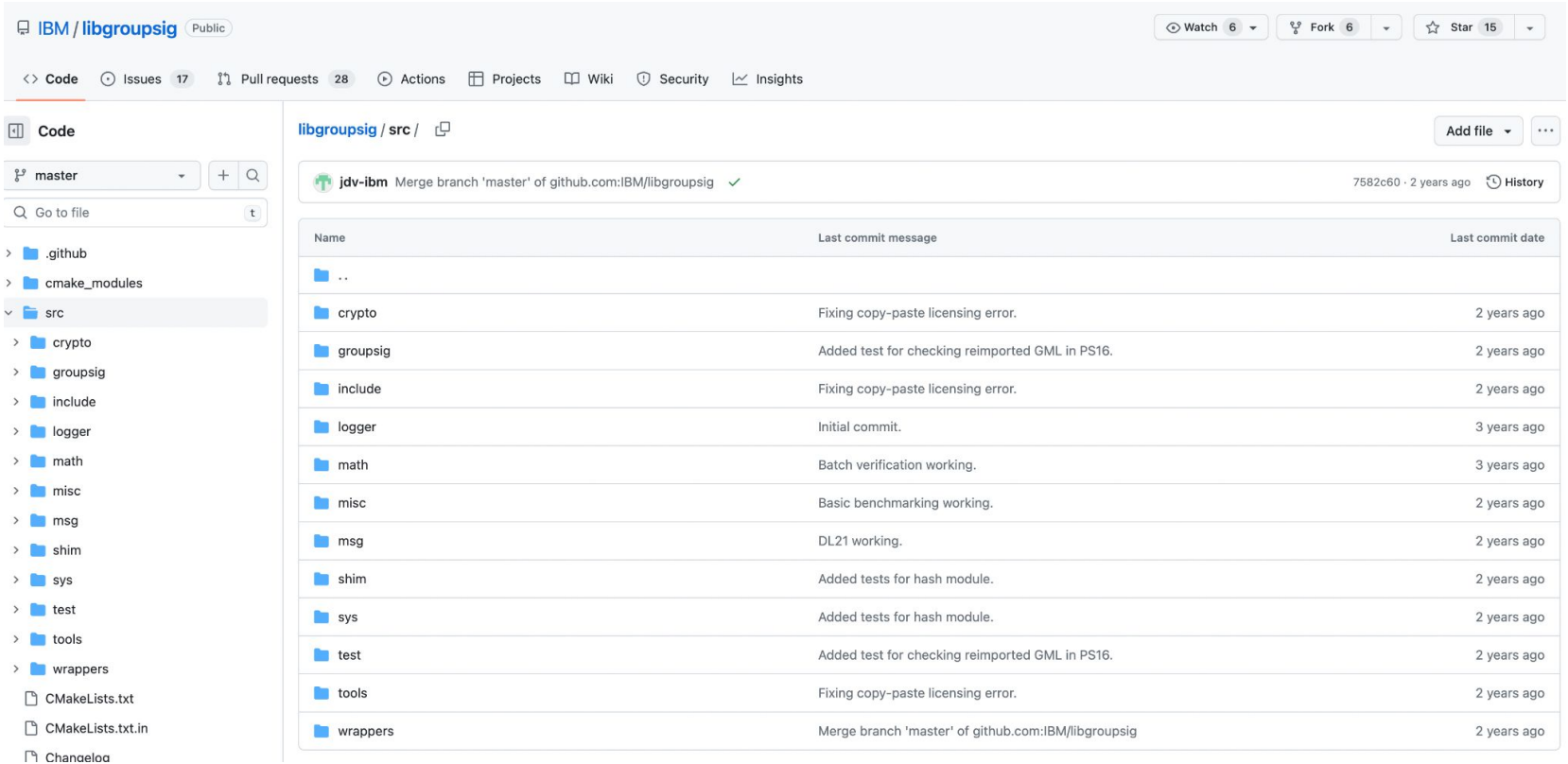
3. Pick random x_i's: Third, the signer picks random x_i for all the other ring members $1 \le i \le r$, $i \ne s$ uniformly and independently from $\{0,1\}^b$, and computes

$$y_i = g_i(x_i) \ .$$

4. Solve for y_s: Fourth, the signer solves the following ring equation for y_s:

$$C_{k,v}(y_1, y_2, \ldots, y_r) = v \ .$$

Context: Group signatures



It’s way easier to do this now!

- zkmessage.xyz was built in a weekend
- Most of the work was on writing the webserver and frontend
- Demo

Setup: Hash functions and keypairs

Hash function: takes in some input, deterministically produces some “random-looking” output that the input can’t be reverse-engineered from.

H(secret) = public commitment

https://emn178.github.io/online-tools/sha256.html

We are going to use (secret, public commitment) as a keypair.

Setup: Hash functions and keypairs

Everyone joining zkmessage.xyz locally generates a (secret, commitment) pair.

New users publish their commitment on Twitter, publicly associating their Twitter handle with their commitment. Because commitment is a hash of secret, secret acts like a private key / password for the account.

v0: group membership

Posting: what's being proven?

Fill in a plain English sentence:

I am one of Alice, Bob, or Charlie

I know either Alice's secret, Bob's secret, or Charlie's secret

I know a number X that hashes to Y1, or Y2, or Y3

I know a number X such that (H(X) - Y1) * (H(X) - Y2) * (H(X) - Y3) = 0

I know a number X, such that Y := H(X), and (Y - Y1) * (Y - Y2) * (Y - Y3) = 0

Posting: what's being proven?

I know:

Private	X, Y
Public	Y1, Y2, Y3

Posting: what's being proven?

	User-provided	Computed
Private		
Public		

Posting: what's being proven?

I know:

	Inputs	Computed
Private	X	Y
Public	Y1, Y2, Y3	

Posting: what's being proven?

I know:

	Inputs	Computed
Private	X	Y, TEMP
Public	Y1, Y2, Y3	

Such that:

Y := H(X)

TEMP := (Y - Y1) * (Y - Y2)

TEMP * (Y - Y3) = 0

v1: group membership + specific message

Posting: what's being proven?

I know:

	Inputs	Computed
Private	X	Y, TEMP
Public	Y1, Y2, Y3, M	

Such that:

Y := H(X)

TEMP := (Y - Y1) * (Y - Y2)

TEMP * (Y - Y3) = 0

v2: group membership + specific message + can prove post facto you said it or didn’t say it

Posting: what's being proven?

I know:

	Inputs	Computed
Private	X, S	Y, TEMP
Public	Y1, Y2, Y3, M	T

Such that:

Y := H(X)

TEMP := (Y - Y1) * (Y - Y2)

TEMP * (Y - Y3) = 0

T := H(S)

Downsides:

- Have to save S for every message you've ever sent
- Doesn't allow you to disavow a message

Posting: what's being proven?

I know:

	Inputs	Computed
Private	X	Y, TEMP
Public	Y1, Y2, Y3, M	C

Such that:

Y := H(X)

TEMP := (Y - Y1) * (Y - Y2)

TEMP * (Y - Y3) = 0

C := H(M, X)

Post-facto proving I was the author

I know:

	Inputs	Computed
Private	X	
Public	M, Y, C	

Such that:

$Y = H(X)$

$C = H(M, X)$

Post-facto proving I wasn't the author

I know:

	Inputs	Computed
Private	X	
Public	M, Y, C	

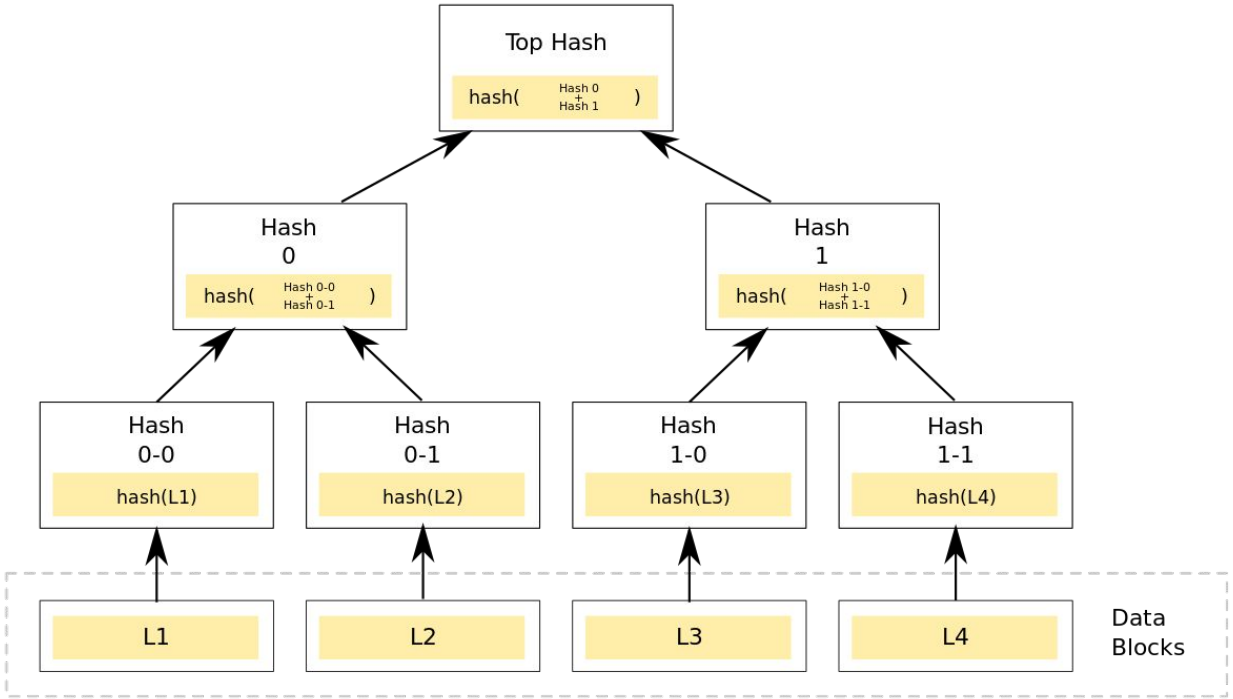
Such that:

Y = H(X)

IsEqual(C, H(M, X)) = 0

v3: group membership + specific message + huge groups

Interlude: Merkle Trees



Merkle Proofs

- A protocol for:
  - Assigning a cryptographic ID to a set (“Merkle root”)
  - Generating short proofs that a given element is in the set
    - Specifically: proving that the element is in *some* hash chain that results in the set’s cryptographic ID

Posting: what's being proven?

I know:

	Inputs	Computed
Private	X, MP	Y
Public	M, GID	

Such that:
Y := H(X)
MERKLE_VERIFY(MP, Y, GID) == 1

Also, stored on / served by the backend:
Some registry that allows the verifier to retrieve (and check) what keys are in any given GID

v4: RSA group membership

double-blind

Public

github.com/gubsheep.keys

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAACAQCh4+DyOTRvrxAAb2W6N4MUf4msUSKhY5/dRezFkwHicxoxdQApev/PsiwUqw3QOWXu8k8CrOrtI8
z+JxHgY3VDAoSnMzWEdJND3RXqgVF62VRGa3H7jJY8zvwq5ScAJ/s3nd1nktP8mOhOZevHFlZm9kYA1Oud9MdXRKm2hvqU2oeNldkEtbLX
atybKONu0rBa0ihGTOptR5dasQZZnBNB2MR6LreSQz/8JzT2YWTq1FCfpVu/9sexR4EE5ozX+sUPKGE9N1ekJgfCakEP5ZSVqHNpdCi7IC
M17BV1D73w55OaUhDxC9bwjnO+iVRrrFnceqP0UzfxKaN6ocOKJ/uCl4nK0LEabKqR5jLo/UUox2/a+IerapEajdAoahuXnvlJQ==
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIBl3E9HuT8nNfv2SI7KYGf0X3dnVLPQra35TMS8xZw/T
```

```
124  // Verify an SSH signature, assuming the public exponent is 65537.
125  // Base message is the DER-encoded hashed message.
126  // Assumes the modulus and base_message are well-formed and range-checked (or
127  // otherwise trustworthy).
128  template RSAVerify65537(n, k) {
```

Agenda

- Preamble: zkSNARKs are “programmable” cryptography
- Part 1: How to use zkSNARKs in apps
  - Group / ring signatures
  - **Anonymous polling / private cryptocurrency**
  - Dark Forest
  - Information marketplaces
- Part 2: Three categories of zkSNARK applications
  - Adding hidden information to decentralized systems
  - Speeding up blockchains
  - Language of Truth

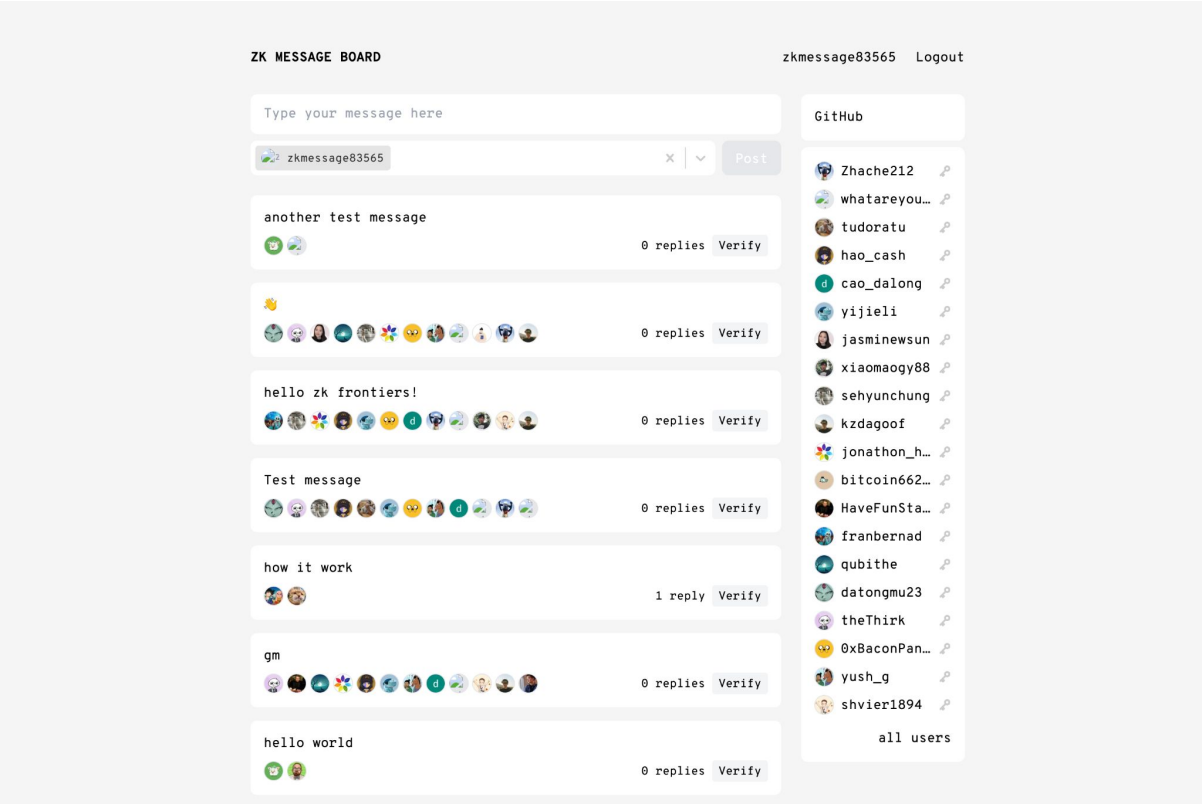
Nullifiers

Building a polling application

Design requirements:

- Server should be able to verify that everyone voting is a member of the voting group (i.e. someone in a known list of public keys)
- Server shouldn’t be able to determine the identity behind any vote
- No one should be able to vote more than once

How to upgrade zkmessage.xyz to polling?



Zupoll


ZUPOLL
Logout
Create
The server never learns your identity.
One vote per Zuzalu participant.
Unlinkable votes across ballots/devices.
Advisory Votes
Official advisory ballots from the Zuzalu organizers
How has your Zuzalu experience been?
Expired
Town Hall 05/16 Longevity/MNE Day Retrospective
Expired
Town Hall 05/09 AIxCrypto Retrospective
Expired
Town Hall 05/02 Network State Retrospective
Expired
Town Hall 04/25 Advisory Vote
Expired
Straw Polls
Unofficial ballots from all Zuzalu residents
Are you vaxxed?
Expires in <105 days
Do you still check Zupoll and Zucast?
Expires in <23 days

Zupoll demo

Proposal

ZKMessage, but there’s a question at the top, the group is locked, and all the messages are **(vote_id)** (i.e. 0 or 1).

Is this enough?

Posting: what's being proven?

Fill in a plain English sentence:

I am one of Alice, Bob, or Charlie[, and I haven't voted yet]

I know either Alice's secret, Bob's secret, or Charlie's secret [, and I haven't voted yet]

I know a number X that hashes to Y1, or Y2, or Y3[, and I haven't voted yet]

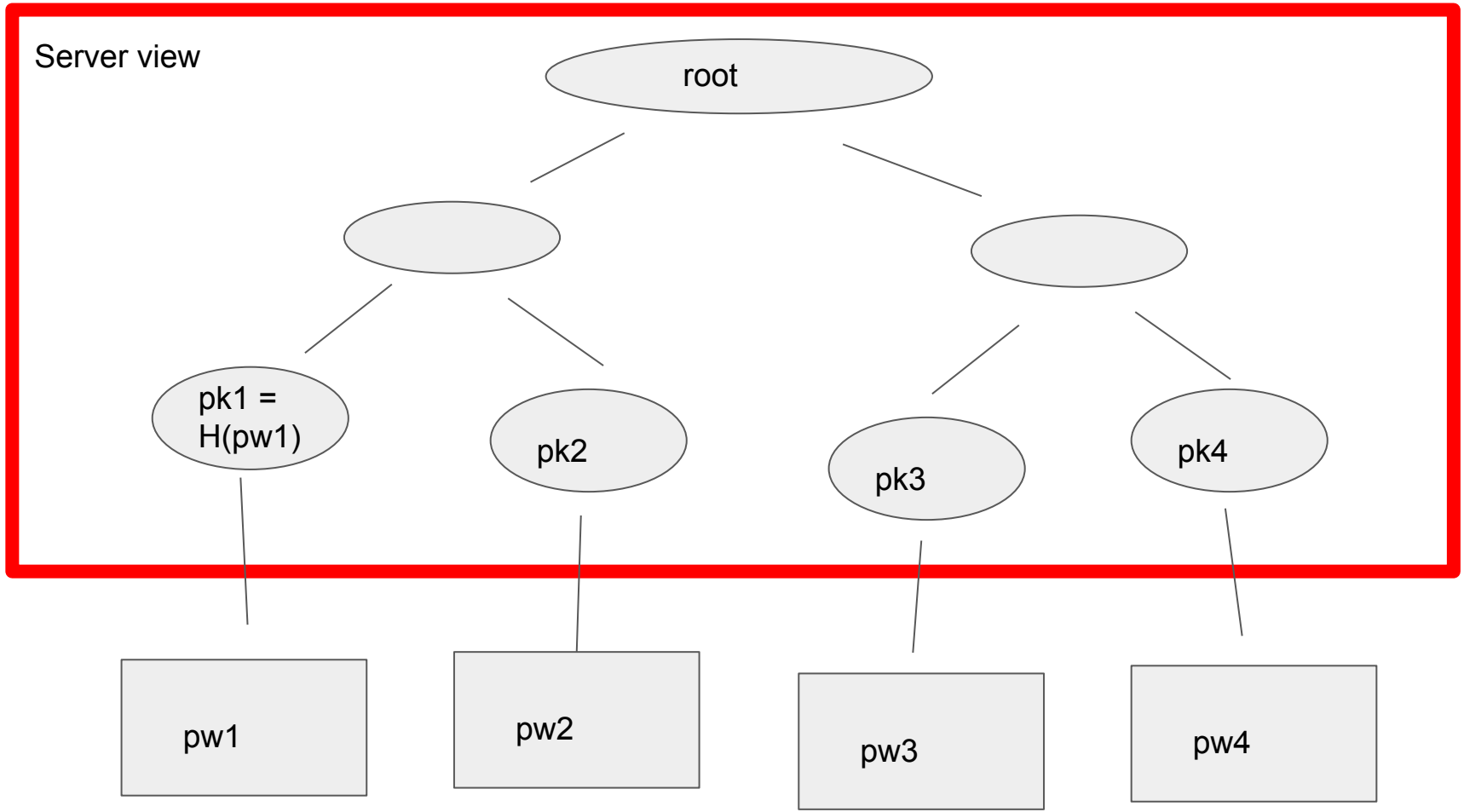
I know a number X such that (H(X) - Y1) * (H(X) - Y2) * (H(X) - Y3) = 0[, and I haven't voted yet]

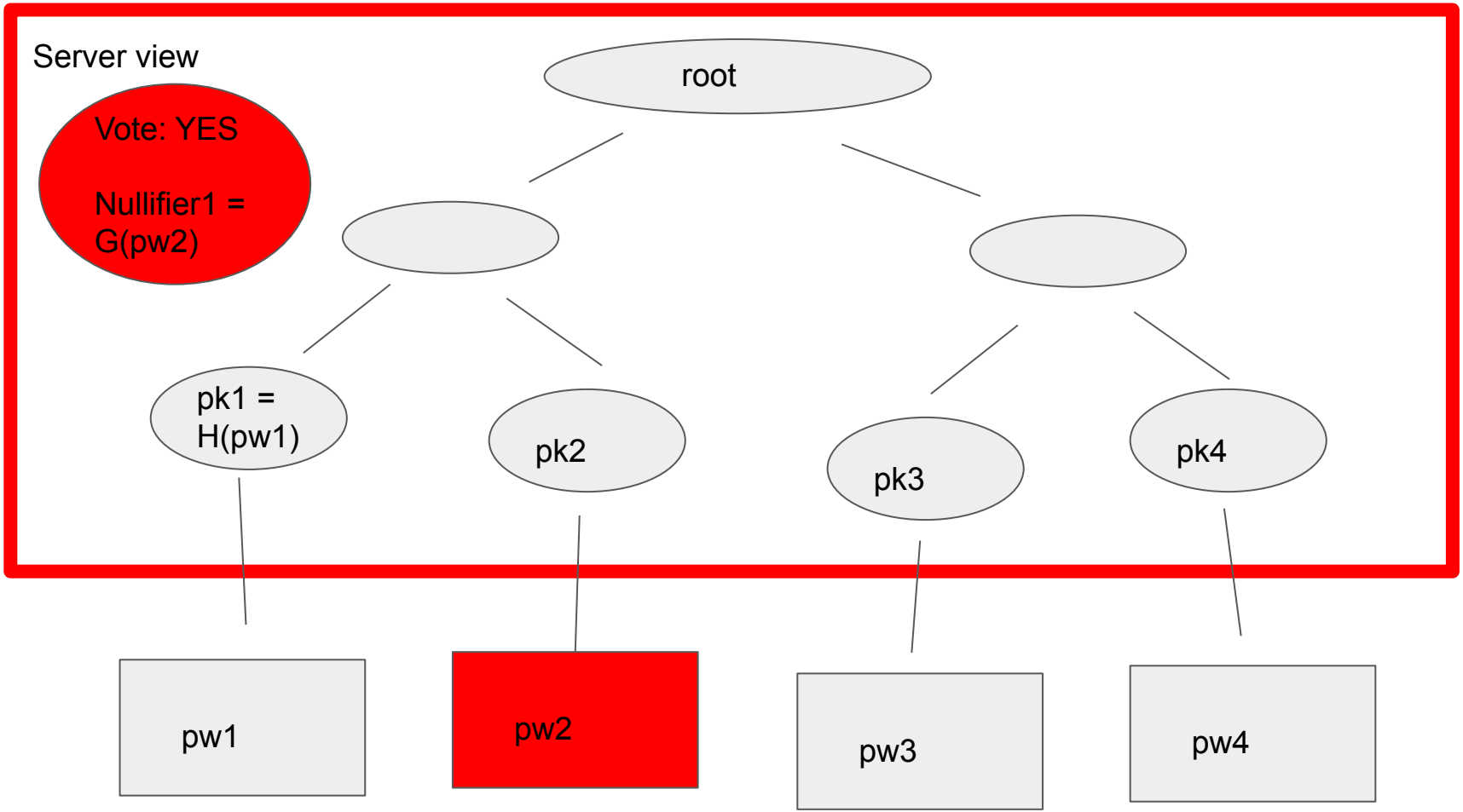
I know a number X, such that Y := H(X), and (Y - Y1) * (Y - Y2) * (Y - Y3) = 0[, and I haven't voted yet]

Idea: nullifiers

Compute and share a deterministic poster ID that is unlinkable to the original poster.

If someone posts twice, you won’t know who they are, but you’ll see the same deterministic poster ID come up.


Server view
root
pk1 =
H(pw1)
pk2
pk3
pk4
pw1
pw2
pw3
pw4


Server view
root
Vote: YES
Nullifier1 = G(pw2)
pk1 = H(pw1)
pk2
pk3
pk4
pw1
pw2
pw3
pw4

Posting: what's being proven?

I know:

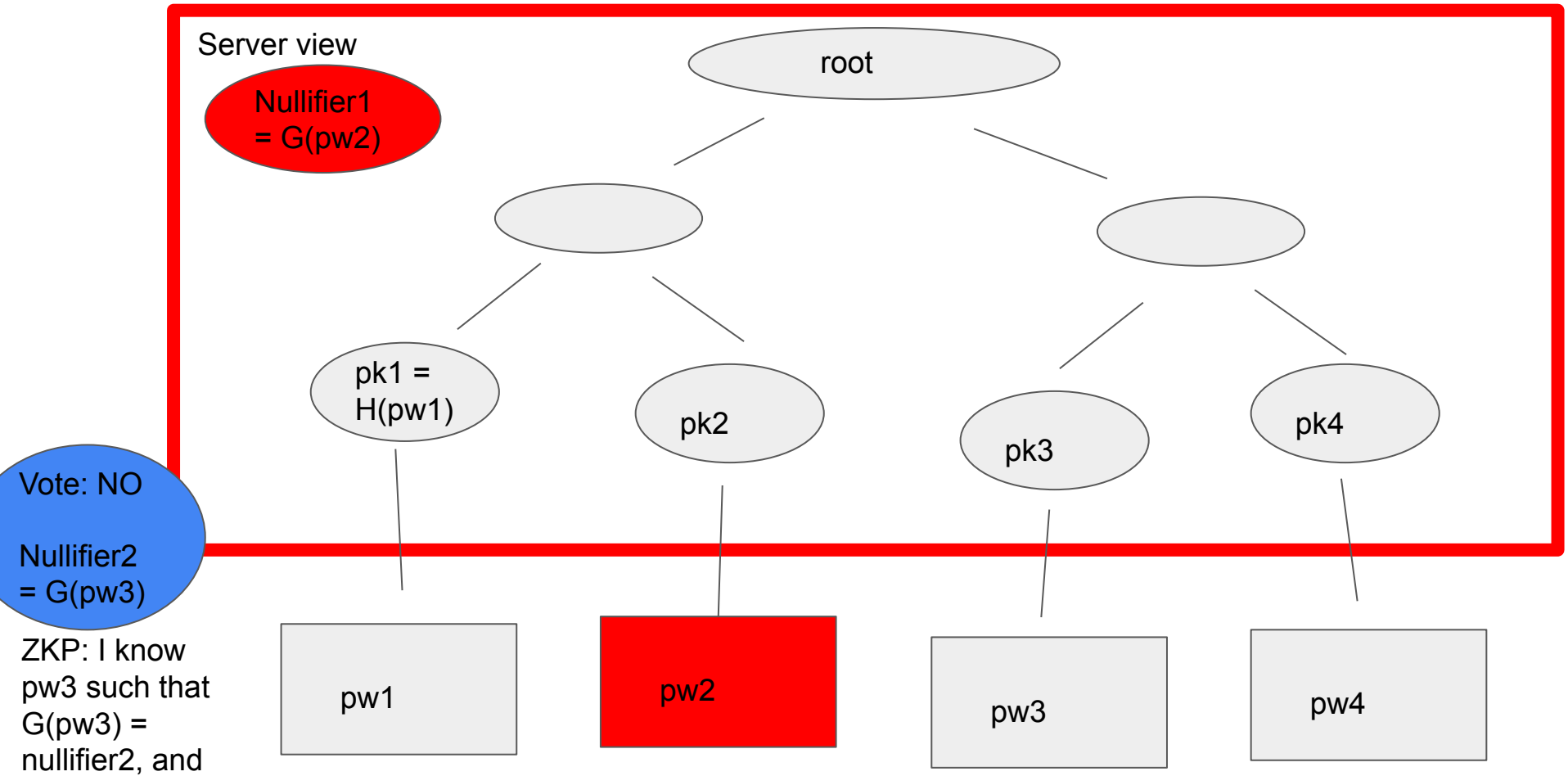
	Inputs	Computed
Private	X, MP	Y, TEMP
Public	GID, V	Z

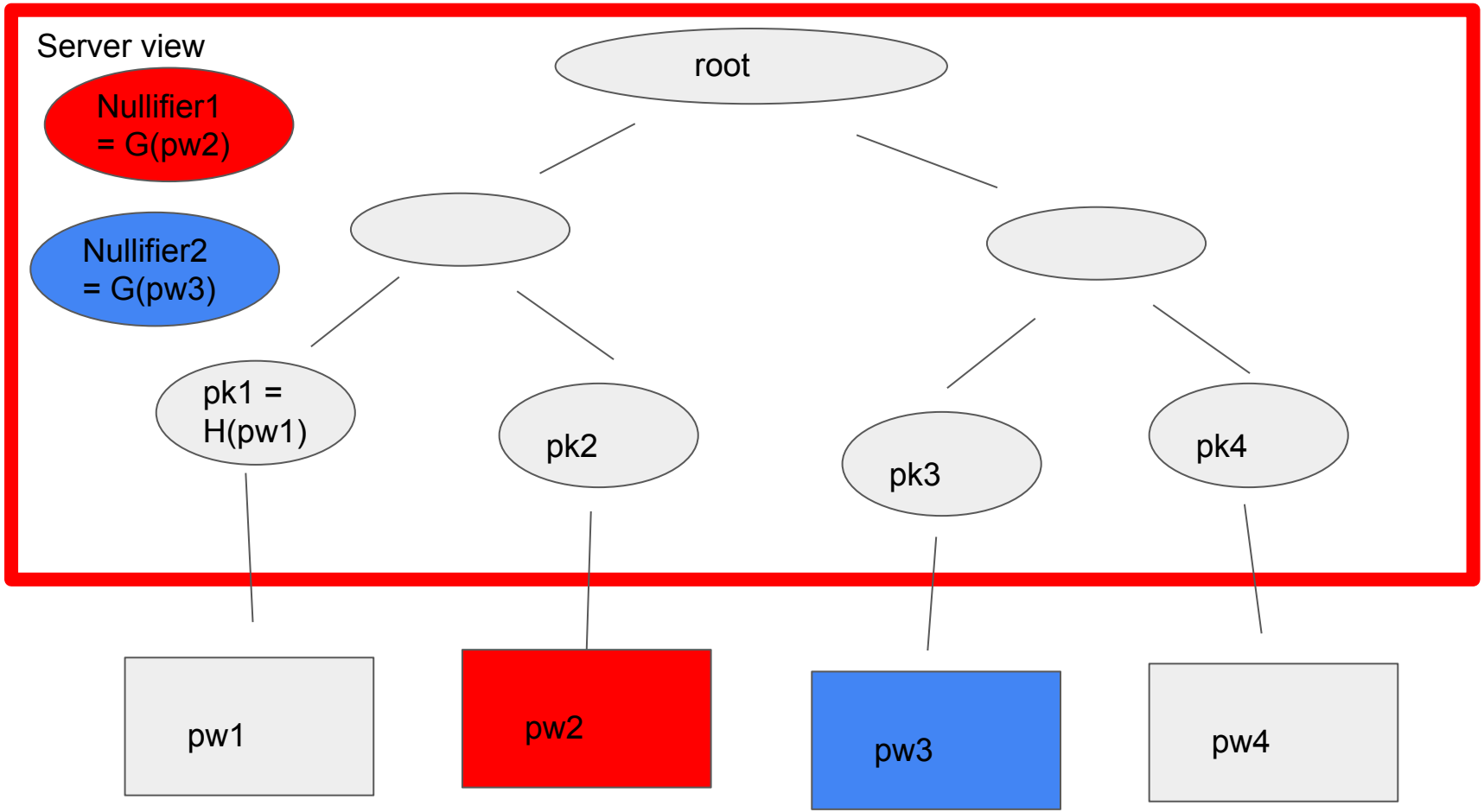
Such that:

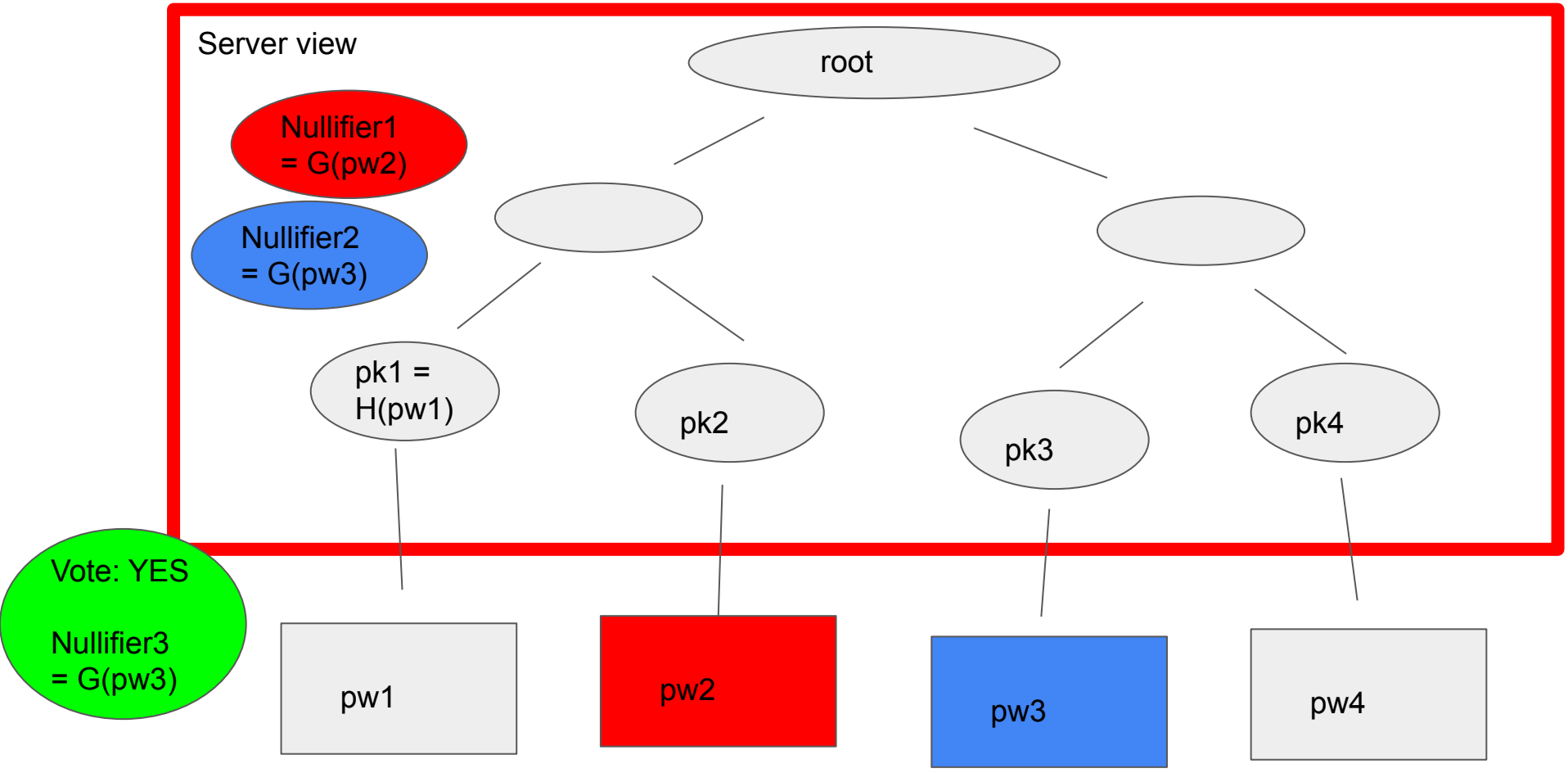
Y := H(X)

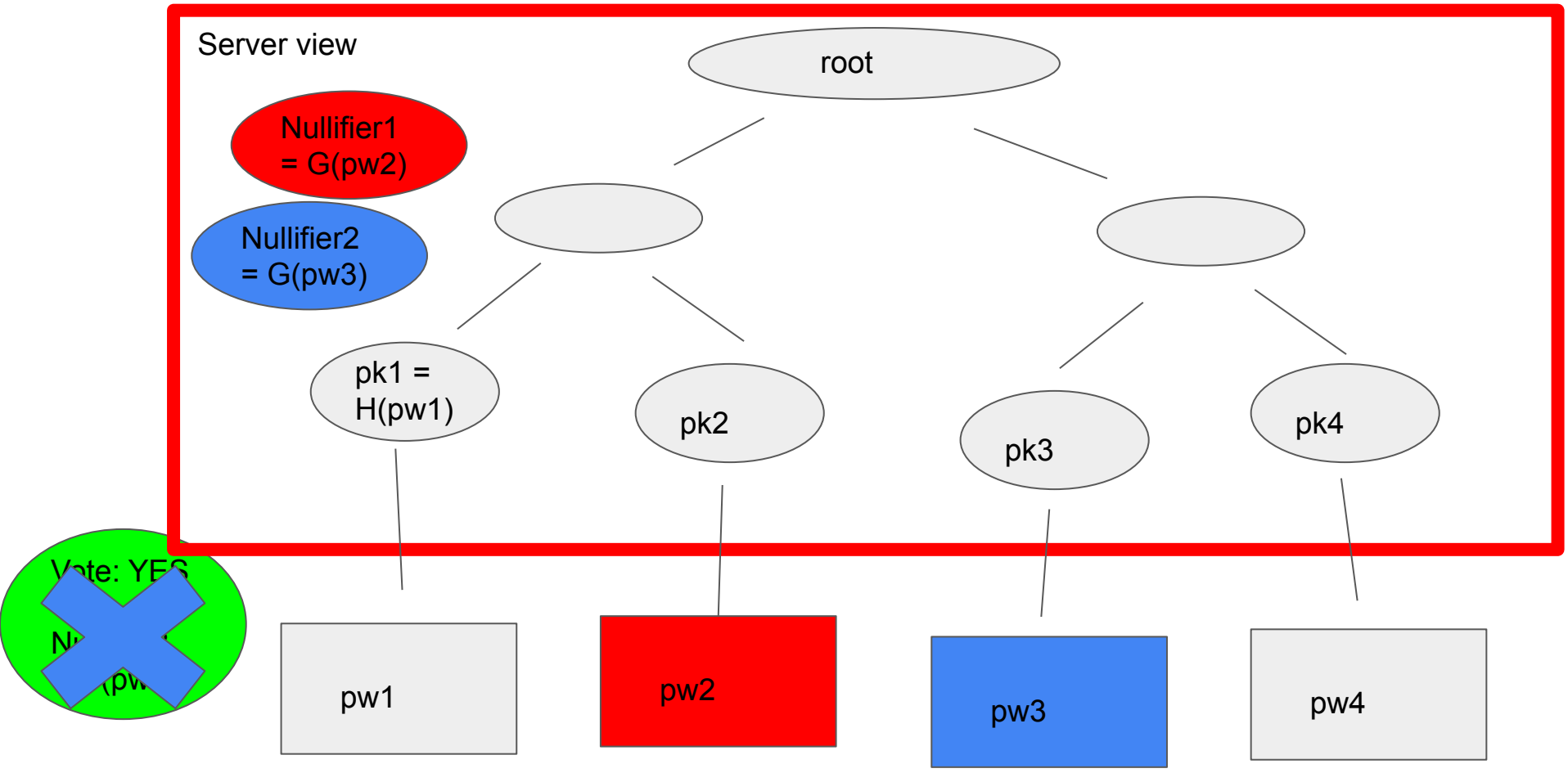
Z := G(X)

MERKLE_VERIFY(MP, Y, GID) == 1


Server view
Nullifier1
= G(pw2)
root
pk1 =
H(pw1)
pk2
pk3
pk4
Vote: NO
Nullifier2
= G(pw3)
ZKP: I know pw3 such that G(pw3) = nullifier2, and H(pw3) is in merkle tree
pw1
pw2
pw3
pw4


Server view
root
Nullifier1
= G(pw2)
Nullifier2
= G(pw3)
pk1 =
H(pw1)
pk2
pk3
pk4
pw1
pw2
pw3
pw4


Server view
root
Nullifier1
= G(pw2)
Nullifier2
= G(pw3)
pk1 =
H(pw1)
pk2
pk3
pk4
Vote: YES
Nullifier3
= G(pw3)
pw1
pw2
pw3
pw4


Server view
root
Nullifier1
= G(pw2)
Nullifier2
= G(pw3)
pk1 =
H(pw1)
pk2
pk3
pk4
pw1
pw2
pw3
pw4

Other usages of nullifiers

Zkmessage: anonymity to pseudonymity

Give everyone a persistent, pseudonymous identity.

- ID can be linked across messages.

ZCash and TornadoCash

Same trick can be used for anonymous digital currency systems.

Goal: make digital currency transfers without having to reveal who sender and recipient are.

Idea: Bitcoin mixers

Someone operates a Bitcoin address and says:

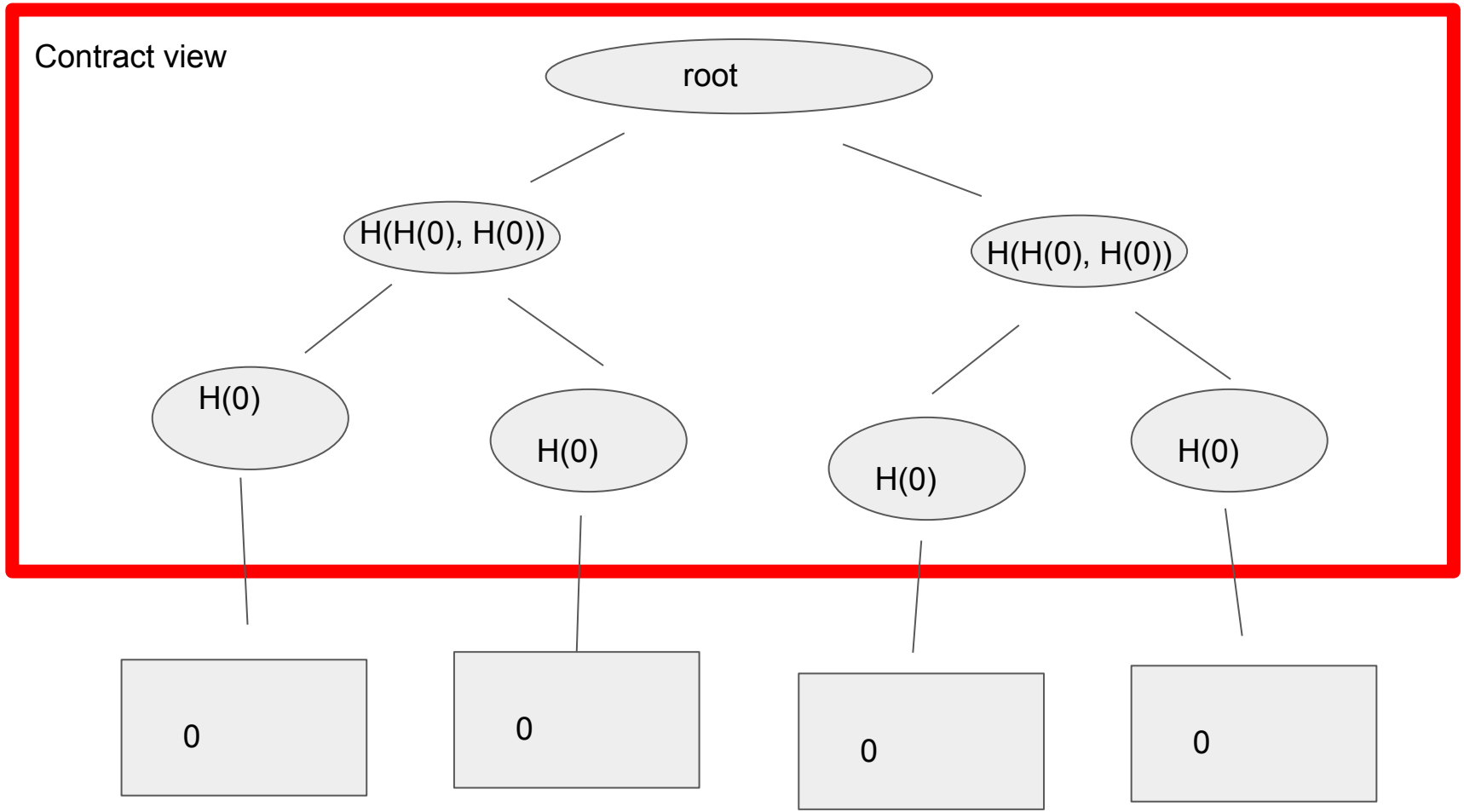
“send Bitcoins here and tell me out-of-band who you want me to send it to. I’ll keep track of everyone’s intended recipients and make all the transfers at once, every 24 hours”

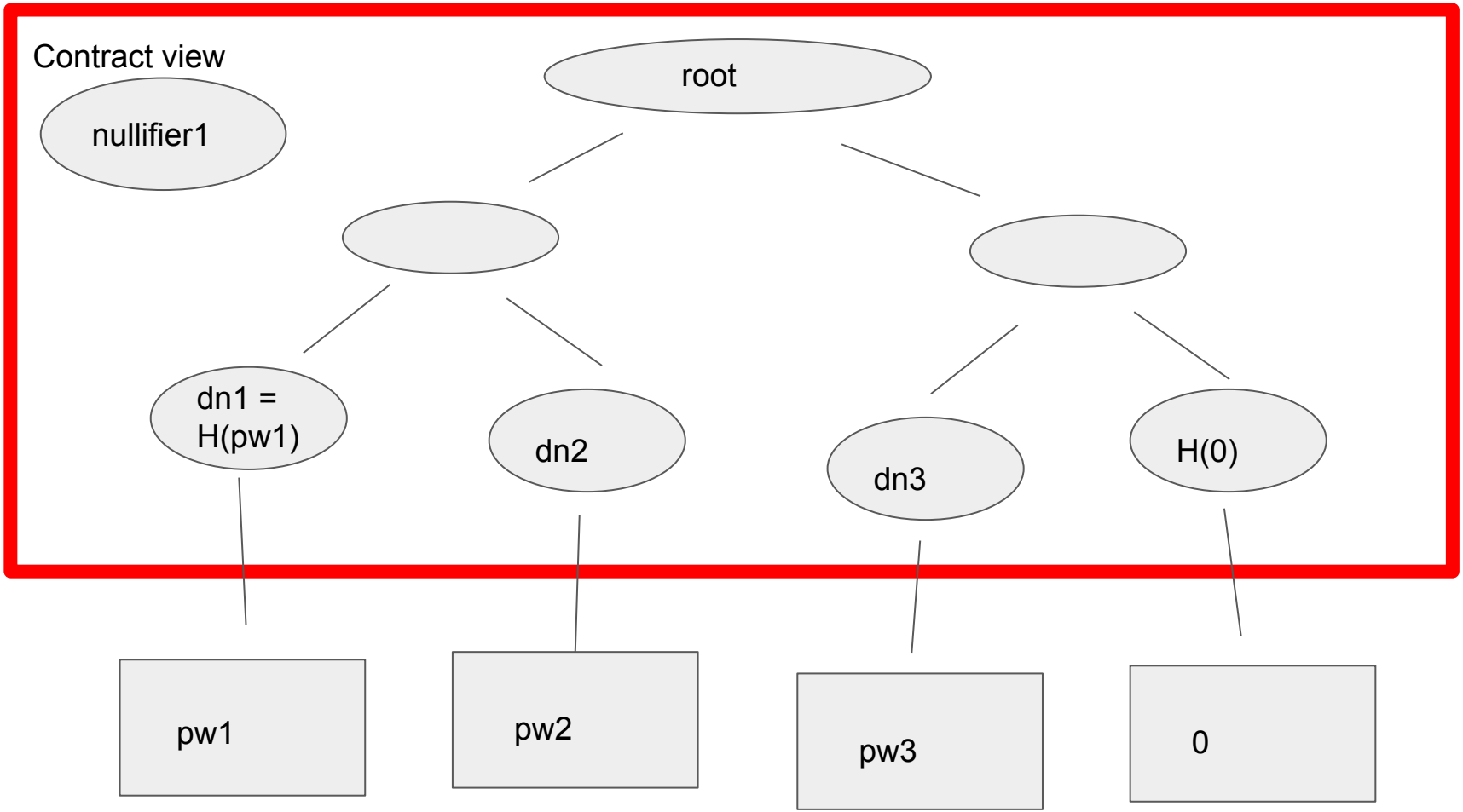
Idea: Bitcoin mixers

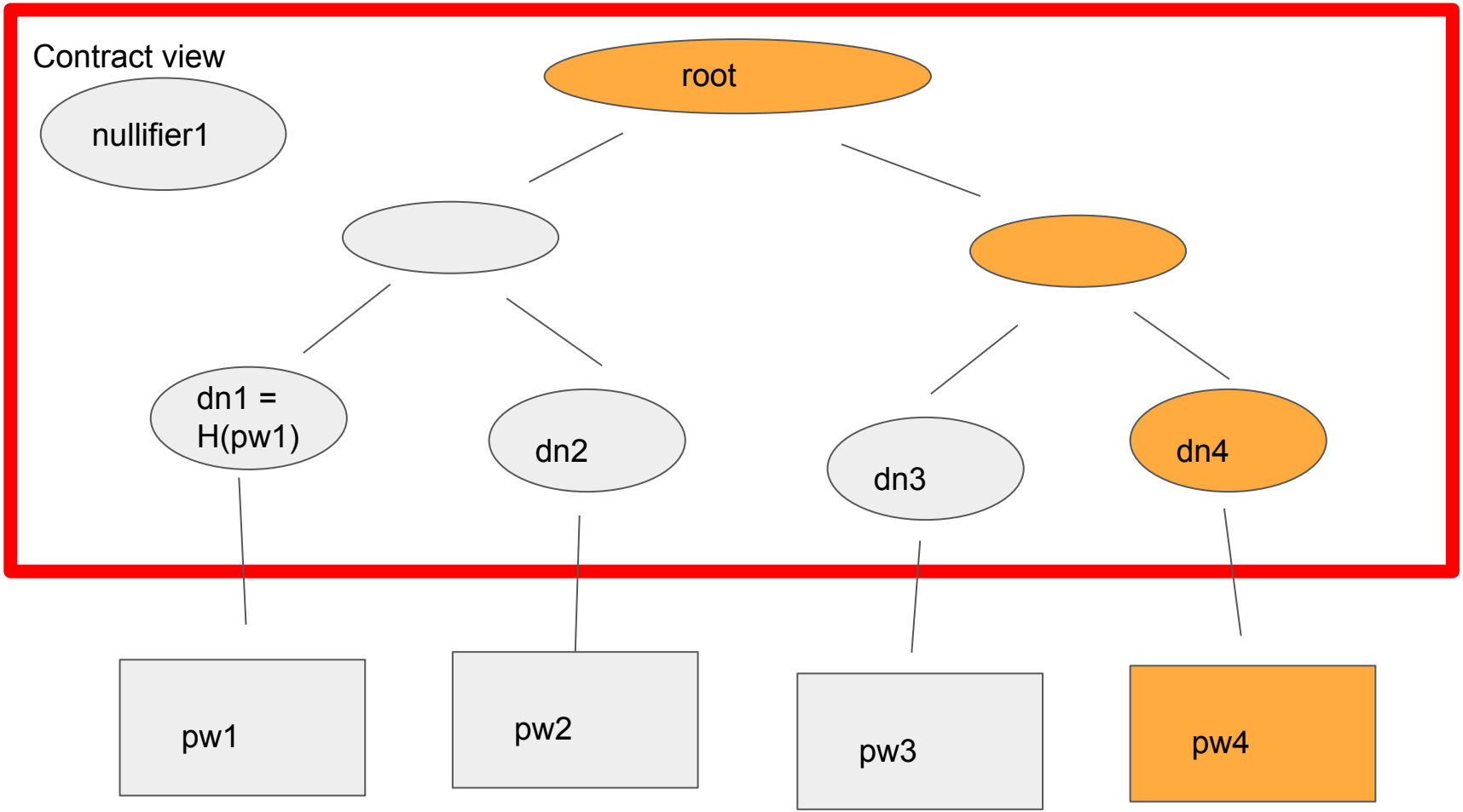
Someone operates a Bitcoin address and says:

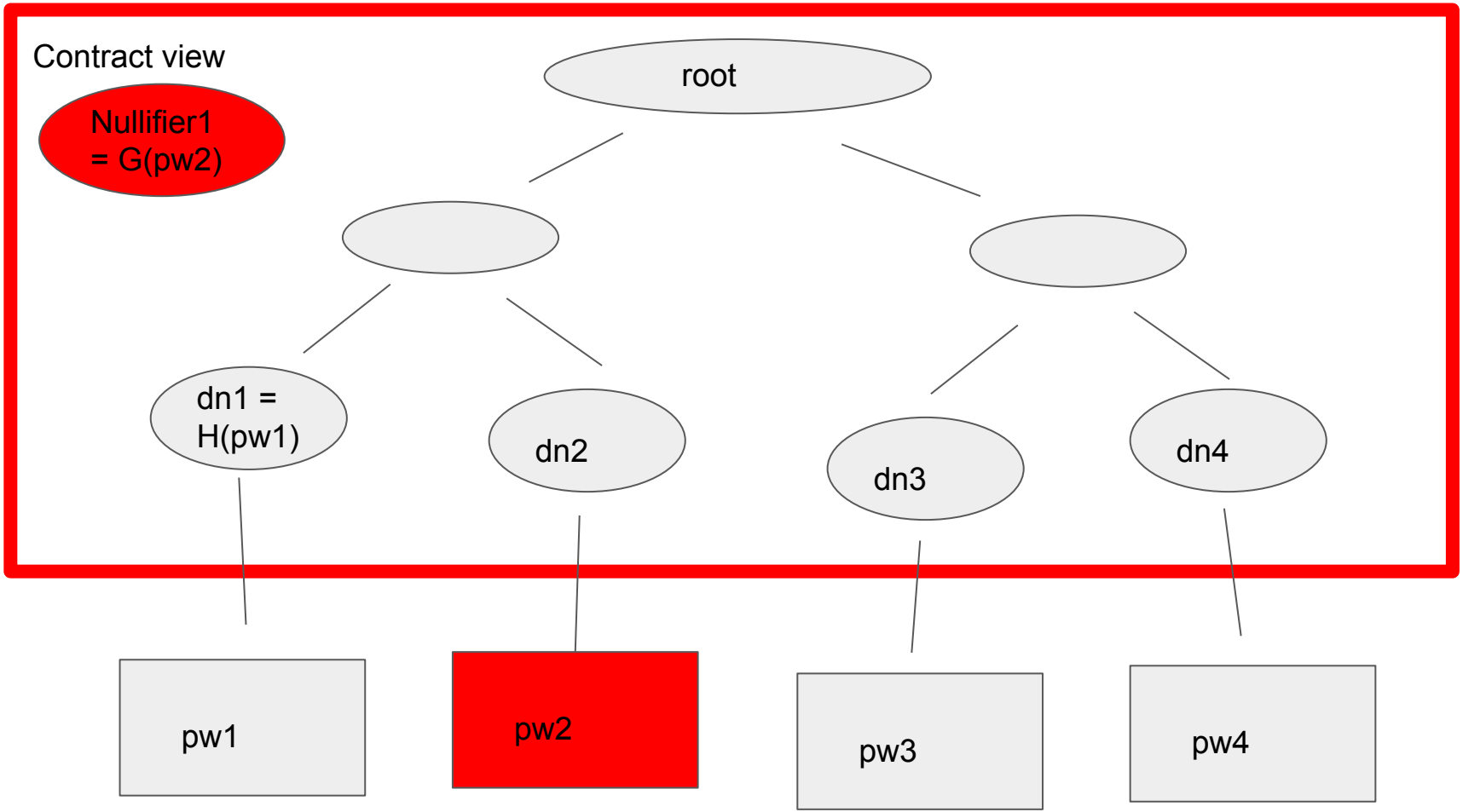
“send Bitcoins here and tell me out-of-band who you want me to send it to. I’ll keep track of everyone’s intended recipients and make all the transfers at once, every 24 hours”

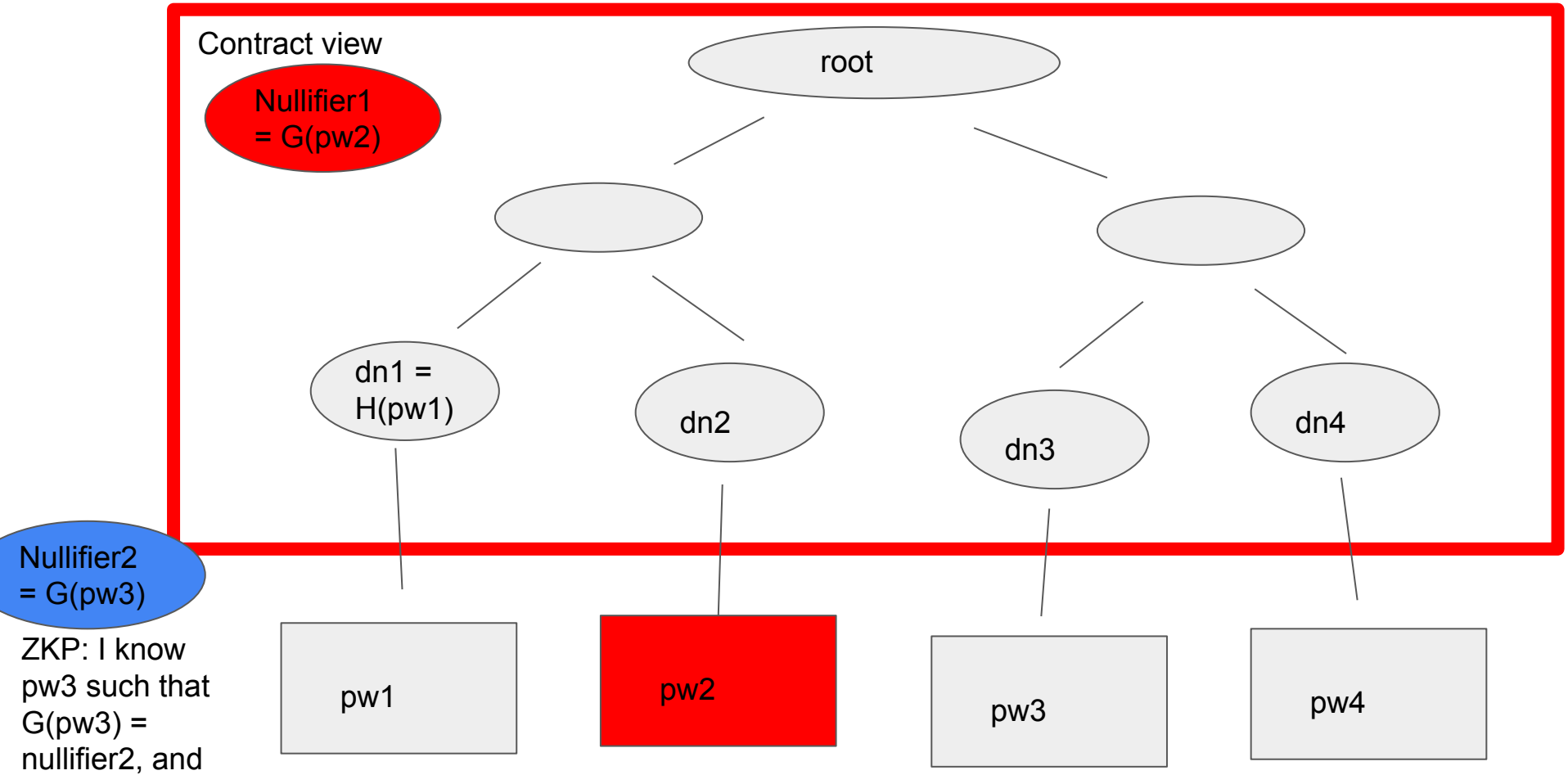
Failure modes?


Contract view
root
H(H(0), H(0))
H(H(0), H(0))
H(0)
H(0)
H(0)
H(0)
0
0
0
0

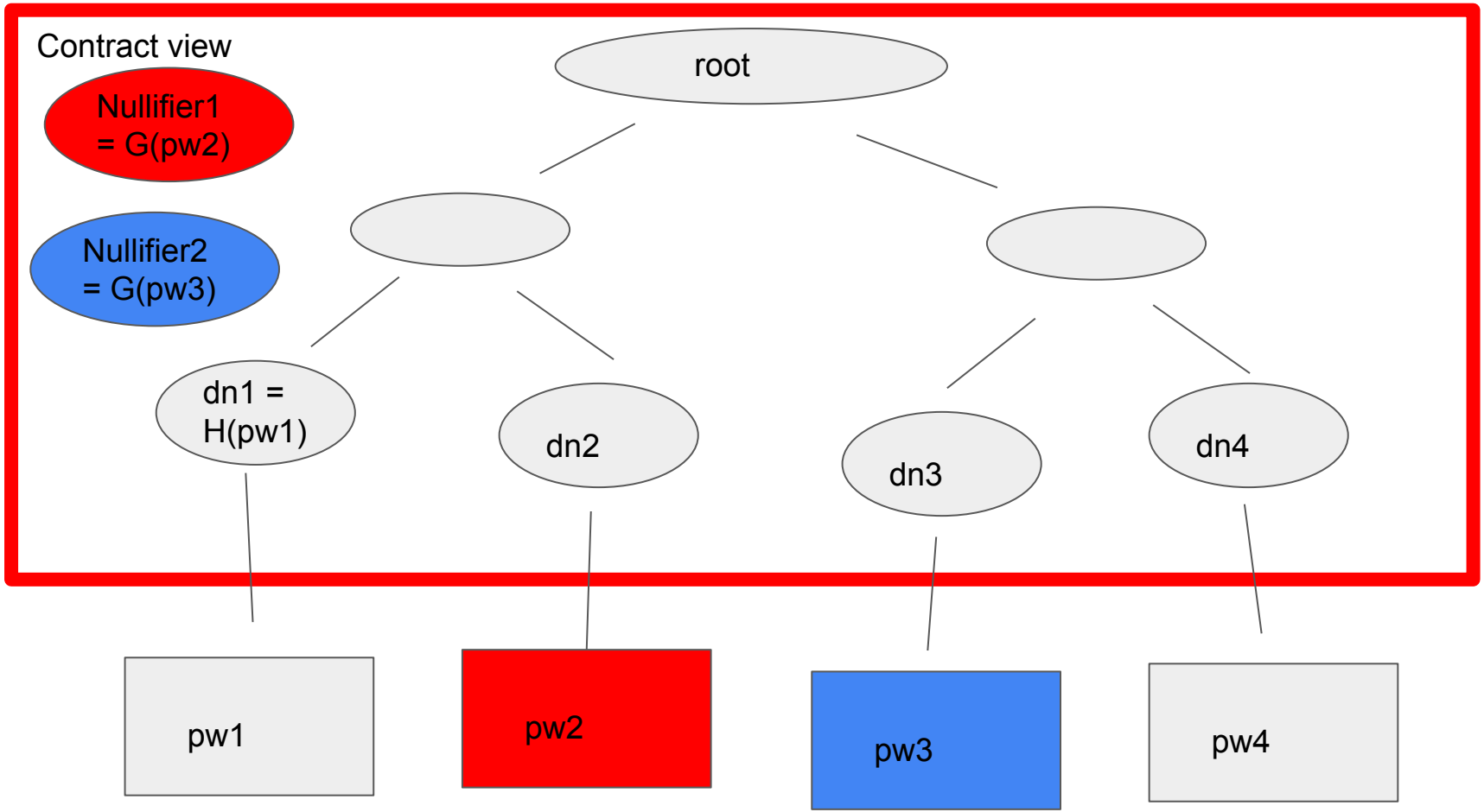

Contract view
root
nullifier1
dn1 =
H(pw1)
dn2
dn3
H(0)
pw1
pw2
pw3
0

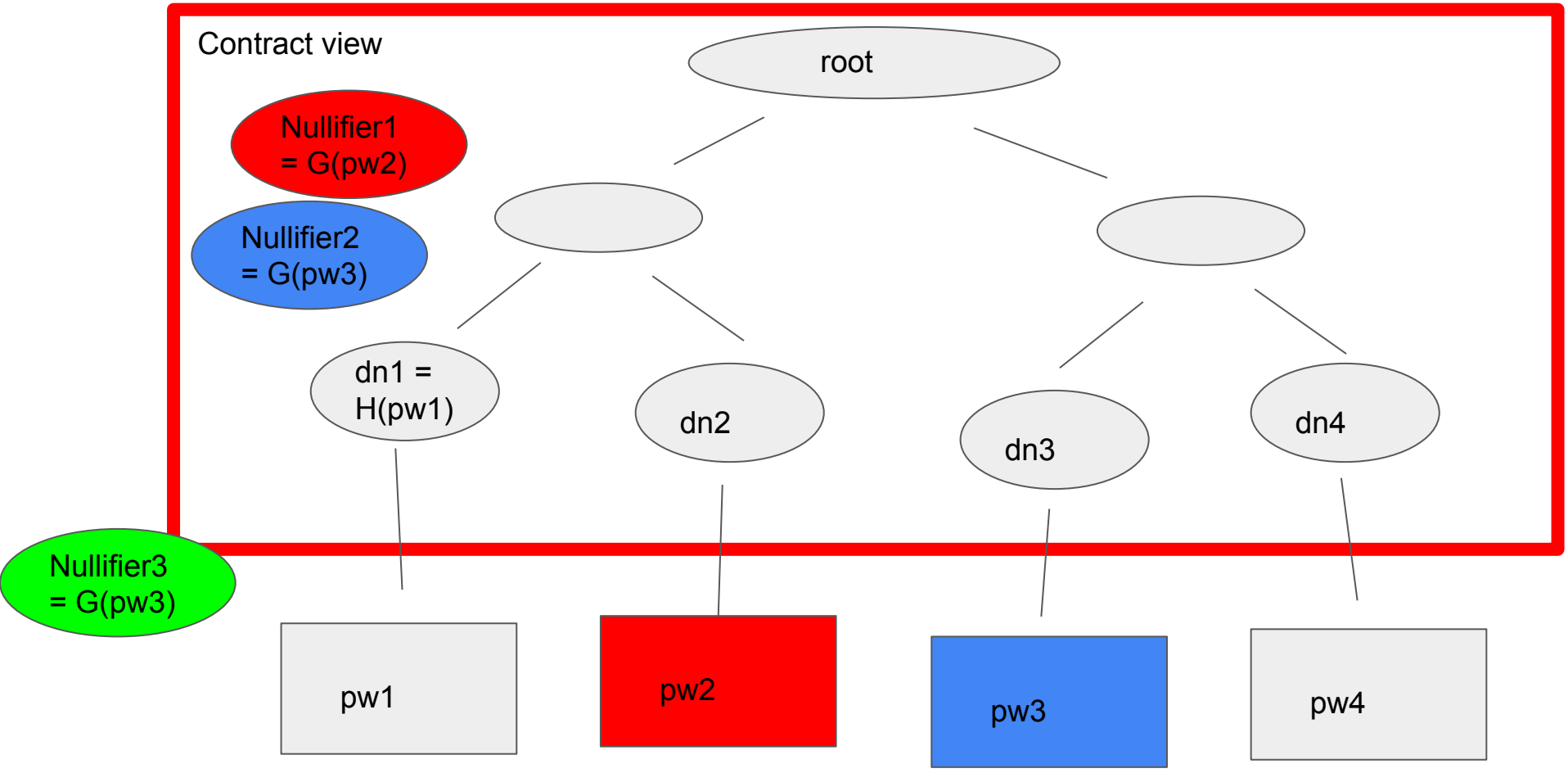

Contract view
root
nullifier1
dn1 =
H(pw1)
dn2
dn3
dn4
pw1
pw2
pw3
pw4

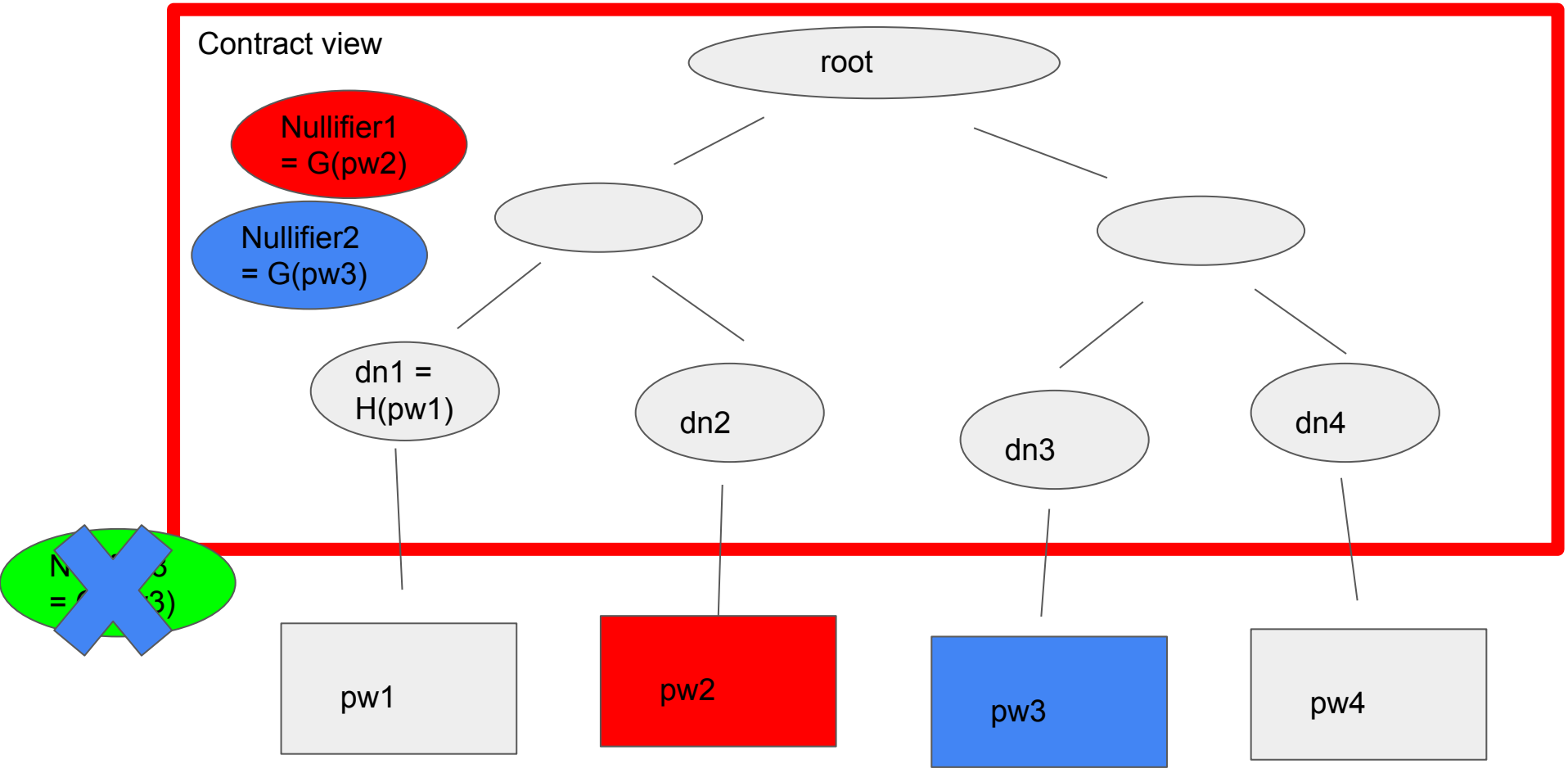

Contract view
Nullifier1
= G(pw2)
root
dn1 =
H(pw1)
dn2
dn3
dn4
pw1
pw2
pw3
pw4



ZKP: I know pw3 such that G(pw3) = nullifier2, and H(pw3) is in merkle tree


Contract view
root
Nullifier1
= G(pw2)
Nullifier2
= G(pw3)
dn1 =
H(pw1)
dn2
dn3
dn4
pw1
pw2
pw3
pw4


Contract view
root
Nullifier1
= G(pw2)
Nullifier2
= G(pw3)
dn1 =
H(pw1)
dn2
dn3
dn4
Nullifier3
= G(pw3)
pw1
pw2
pw3
pw4


Contract view
root
Nullifier1
= G(pw2)
Nullifier2
= G(pw3)
dn1 =
H(pw1)
dn2
dn3
dn4
pw1
pw2
pw3
pw4

Understanding check

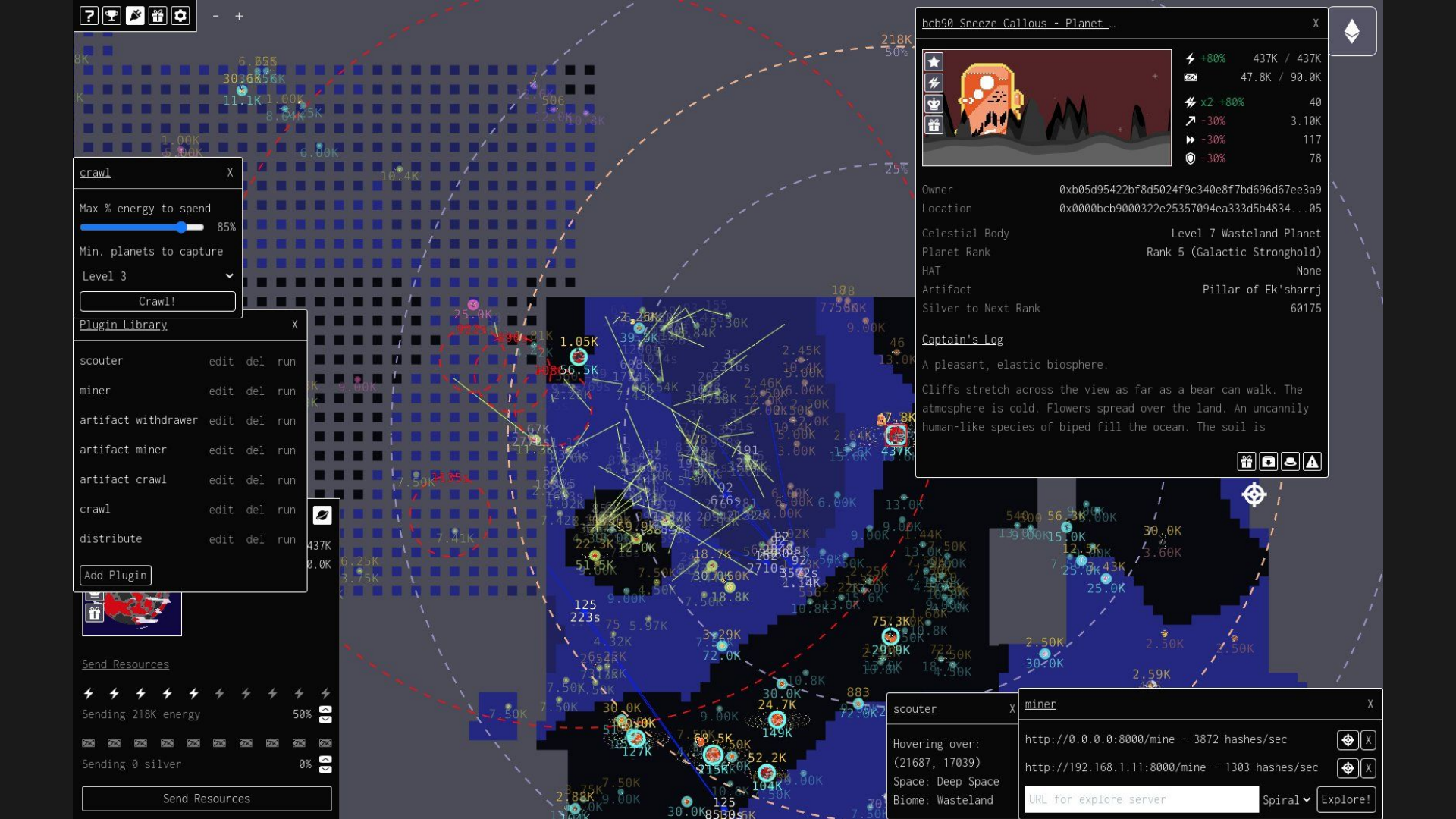
What happens if we make a proof that inputs a value of R that isn’t the actual deposit note tree root?

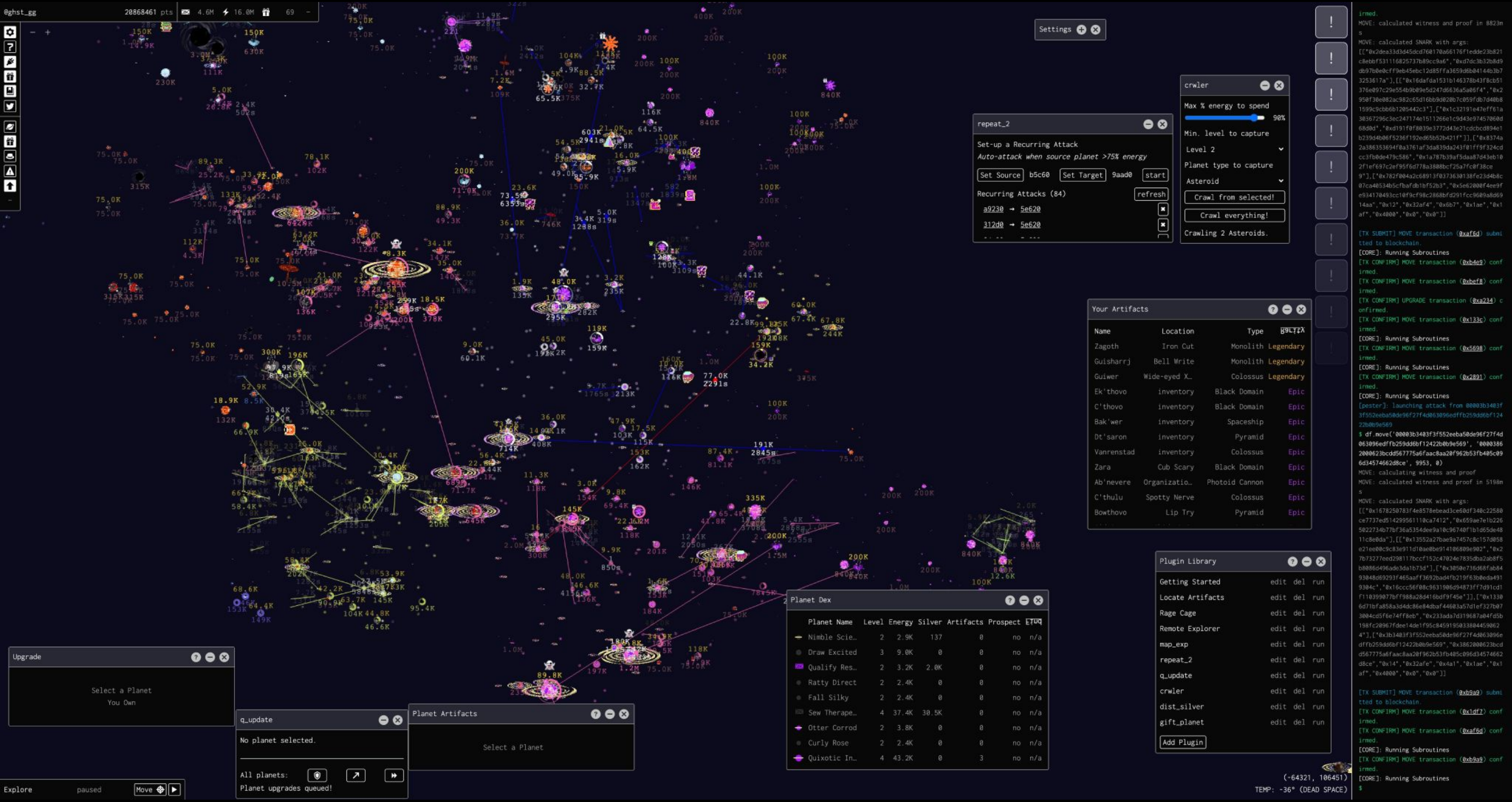
- Will the proof verify?
- Will the protocol break?

Agenda

- Preamble: zkSNARKs are “programmable” cryptography
- Part 1: How to use zkSNARKs in apps
  - Group / ring signatures
  - Anonymous polling / private cryptocurrency
  - **Dark Forest**
  - Information marketplaces
- Part 2: Three categories of zkSNARK applications
  - Adding hidden information to decentralized systems
  - Speeding up blockchains
  - Language of Truth

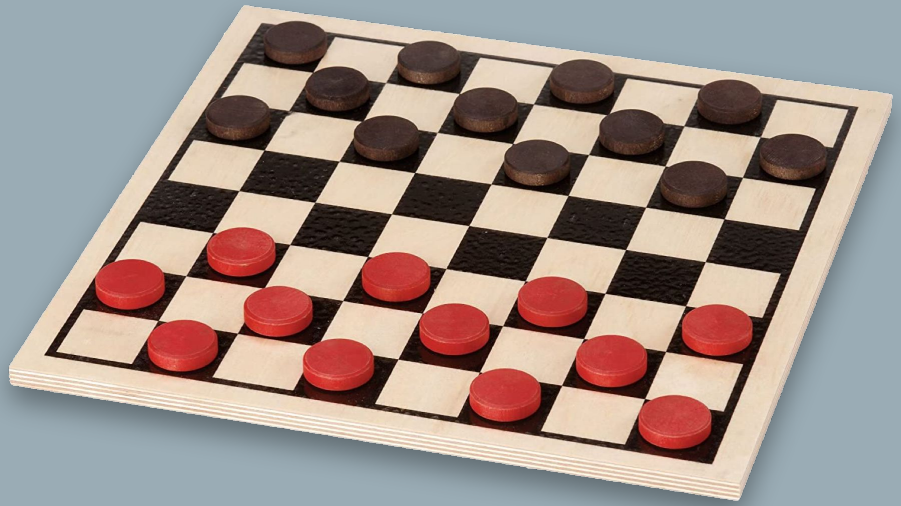

@algoriddims
5506256 pts
1.0M
2.7M
51
gift
Plugin Library
heatmap
Impend Sob
@d_fdao
Lava Planet
Level 4 / Rank 5
77.6K / 93.3K
1.5K / 18.0K
7.77
90
196
3.0K
[TX CONFIRM] PLANET_TRANSFER transaction (0xec4b) confirmed.
[TX SUBMIT] PLANET_TRANSFER transaction (0xd8c9) submitted to blockchain.
[TX CONFIRM] PLANET_TRANSFER transaction (0xd8c9) confirmed.
[TX SUBMIT] PLANET_TRANSFER transaction (0xa6fb) submitted to blockchain.
[TX CONFIRM] PLANET_TRANSFER transaction (0xa6fb) confirmed.
[TX SUBMIT] PLANET_TRANSFER transaction (0x07b0) submitted to blockchain.
[TX CONFIRM] PLANET_TRANSFER transaction (0x07b0) confirmed.
[TX SUBMIT] PLANET_TRANSFER transaction (0xbf57) submitted to blockchain.
[TX CONFIRM] PLANET_TRANSFER transaction (0xbf57) confirmed.
[TX SUBMIT] PLANET_TRANSFER transaction (0xee97) submitted to blockchain.
[TX CONFIRM] PLANET_TRANSFER transaction (0xee97) confirmed.
[TX SUBMIT] PLANET_TRANSFER transaction (0xa5bc) submitted to blockchain.
[TX CONFIRM] PLANET_TRANSFER transaction (0xa5bc) confirmed.
[TX SUBMIT] PLANET_TRANSFER transaction (0x2c91) submitted to blockchain.
[TX CONFIRM] PLANET_TRANSFER transaction (0x2c91) confirmed.
[TX SUBMIT] PLANET_TRANSFER transaction (0xd9e1) submitted to blockchain.
[TX CONFIRM] PLANET_TRANSFER transaction (0xd9e1) confirmed.
[TX SUBMIT] PLANET_TRANSFER transaction (0xd9ec) submitted to blockchain.
[TX CONFIRM] PLANET_TRANSFER transaction (0xd9ec) confirmed.
[TX SUBMIT] PLANET_TRANSFER transaction (0xba4d) submitted to blockchain.
[TX CONFIRM] PLANET_TRANSFER transaction (0xba4d) confirmed.
[TX SUBMIT] PLANET_TRANSFER transaction (0x07a4) submitted to blockchain.
$
303 #/s @ (-187504, 21184)
Move
1 core
(-173880, 15318)
TEMP: 21° (DEEP SPACE)


bcb90 Sneeze Callous - Planet …
+80% 437K / 437K
47.8K / 90.0K
x2 +80% 40
-30% 3.10K
-30% 117
-30% 78
Owner 0xb05d95422bf8d5024f9c340e8f7bd696d67ee3a9
Location 0x0000bcb9000322e25357094ea333d5b4834...05
Celestial Body Level 7 Wasteland Planet
Planet Rank Rank 5 (Galactic Stronghold)
HAT None
Artifact Pillar of Ek'sharrj
Silver to Next Rank 60175
Captain's Log
A pleasant, elastic biosphere.
Cliffs stretch across the view as far as a bear can walk. The atmosphere is cold. Flowers spread over the land. An uncannily human-like species of biped fill the ocean. The soil is
crawl
Max % energy to spend
85%
Min. planets to capture
Level 3
Crawl!
Plugin Library
scouter edit del run
miner edit del run
artifact withdrawer edit del run
artifact miner edit del run
artifact crawl edit del run
crawl edit del run
distribute edit del run
Add Plugin
Send Resources
Sending 218K energy 50%
Sending 0 silver 0%
Send Resources
scouter
Hovering over: (21687, 17039)
Space: Deep Space
Biome: Wasteland
miner
http://0.0.0.0:8000/mine - 3872 hashes/sec
http://192.168.1.11:8000/mine - 1303 hashes/sec
URL for explore server
Spiral
Explore!

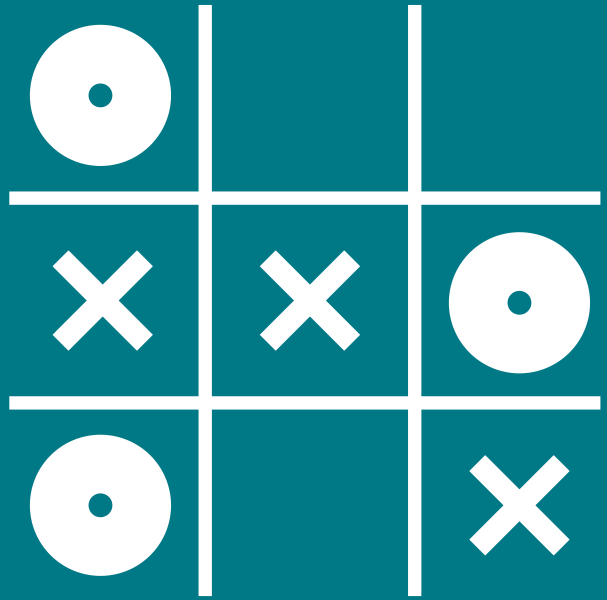

@ghst_gg
20868461 pts
4.6M
16.0M
69
Settings
repeat_2
Set-up a Recurring Attack
Auto-attack when source planet >75% energy
Set Source b5c60 Set Target 9aad0 start
Recurring Attacks (84) refresh
a9230 → 5e620
312d0 → 5e620
crwler
Max % energy to spend
90%
Min. level to capture
Level 2
Planet type to capture
Asteroid
Crawl from selected!
Crawl everything!
Crawling 2 Asteroids.
Your Artifacts
Name Location Type
Zagoth Iron Cut Monolith Legendary
Guisharrj Bell Write Monolith Legendary
Guiwer Wide-eyed X.. Colossus Legendary
Ek'thovo inventory Black Domain Epic
C'thovo inventory Black Domain Epic
Bak'wer inventory Spaceship Epic
Dt'saron inventory Pyramid Epic
Vanrenstad inventory Colossus Epic
Zara Cub Scary Black Domain Epic
Ab'nevere Organizatio.. Photoid Cannon Epic
C'thulu Spotty Nerve Colossus Epic
Bowthovo Lip Try Pyramid Epic
Plugin Library
Getting Started edit del run
Locate Artifacts edit del run
Rage Cage edit del run
Remote Explorer edit del run
map_exp edit del run
repeat_2 edit del run
q_update edit del run
crwler edit del run
dist_silver edit del run
gift_planet edit del run
Add Plugin
Planet Dex
Planet Name Level Energy Silver Artifacts Prospect
Nimble Scie.. 2 2.9K 137 0 no n/a
Draw Excited 3 9.0K 0 0 no n/a
Qualify Res.. 2 3.2K 2.0K 0 no n/a
Ratty Direct 2 2.4K 0 0 no n/a
Fall Silky 2 2.4K 0 0 no n/a
Sew Therape.. 4 37.4K 30.5K 0 no n/a
Otter Corrod 2 3.8K 0 0 no n/a
Curly Rose 2 2.4K 0 0 no n/a
Quixotic In.. 4 43.2K 0 3 no n/a
Upgrade
Select a Planet
You Own
q_update
No planet selected.
All planets:
Planet upgrades queued!
Planet Artifacts
Select a Planet
Explore paused Move
(-64321, 106451)
TEMP: -36° (DEAD SPACE)
[CORE]: Running Subroutines

Why do you need ZK to build a (complex) decentralized game?

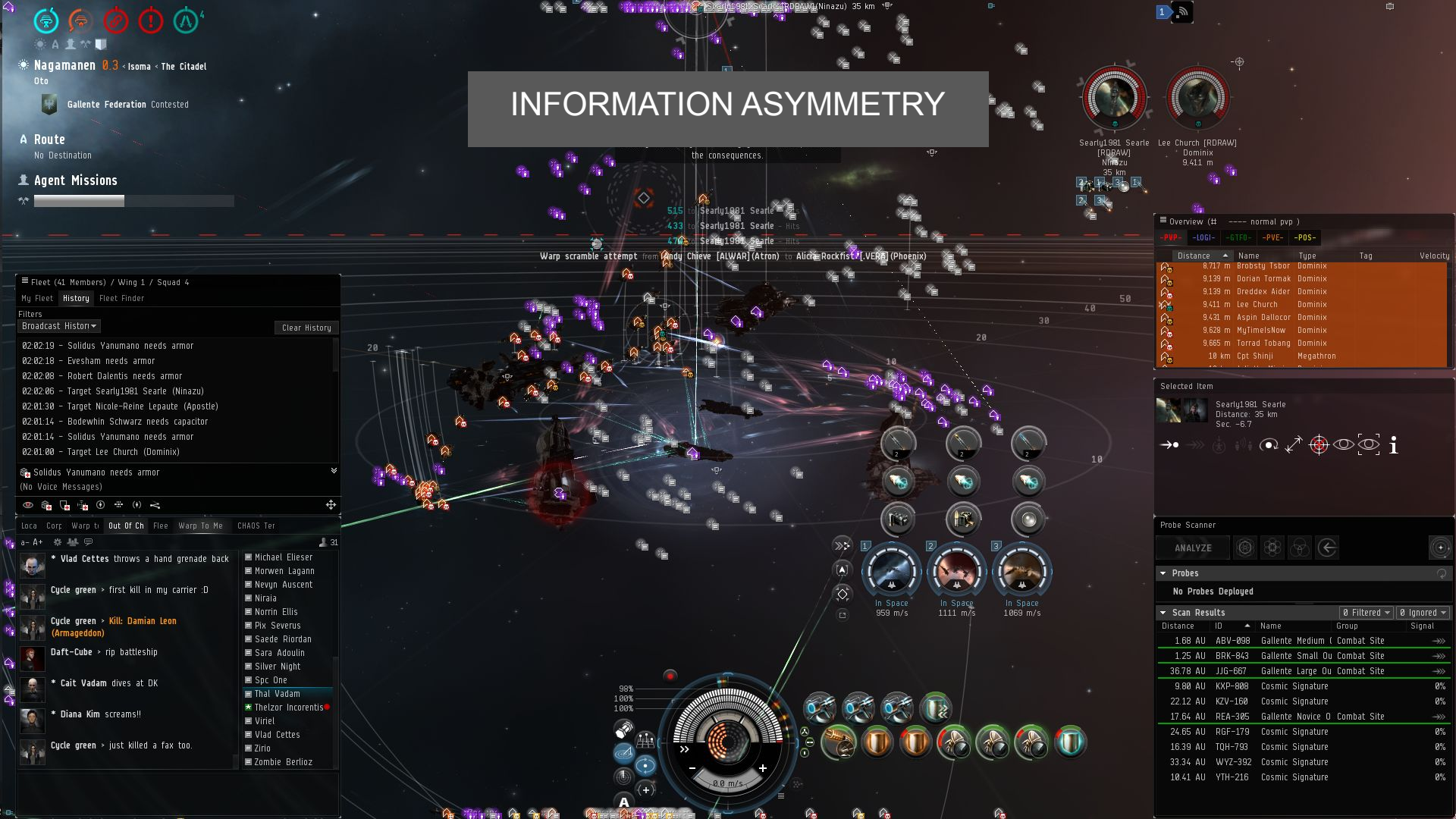
MOTIVATION: COMPLETE AND INCOMPLETE INFORMATION



COMPLETE INFORMATION GAMES

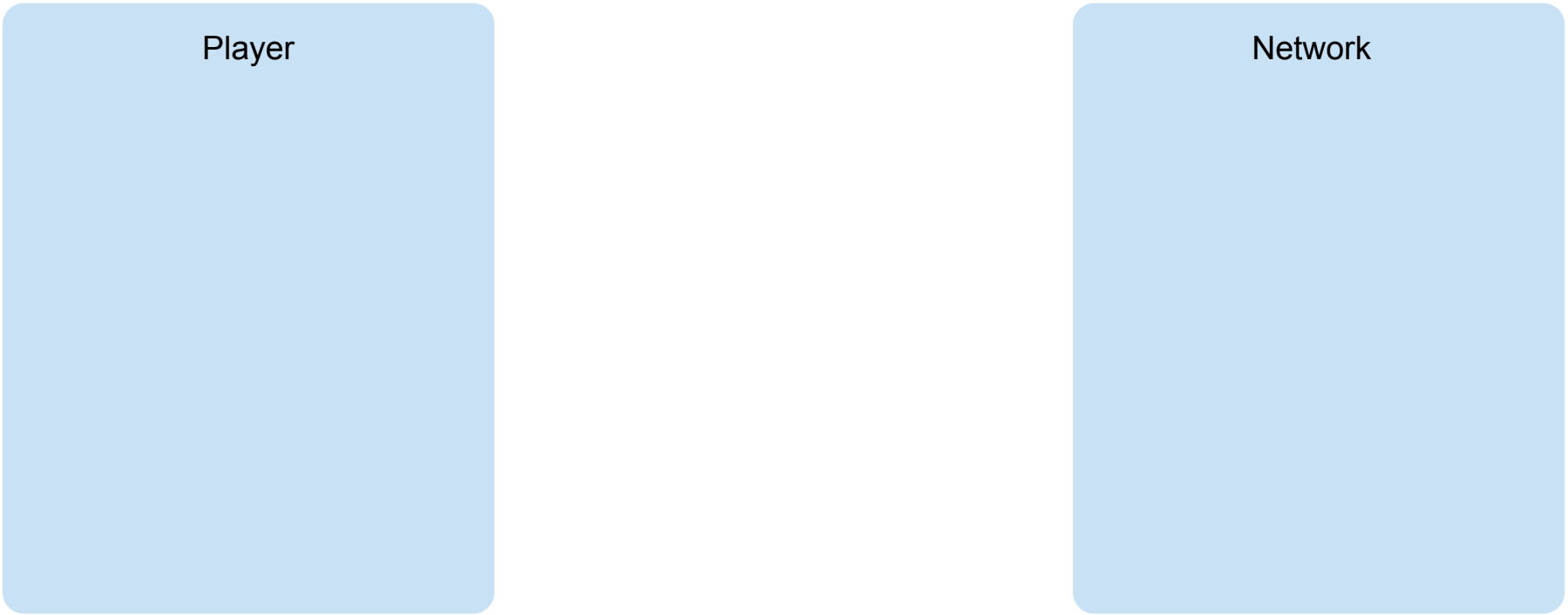


INFORMATION ASYMMETRY

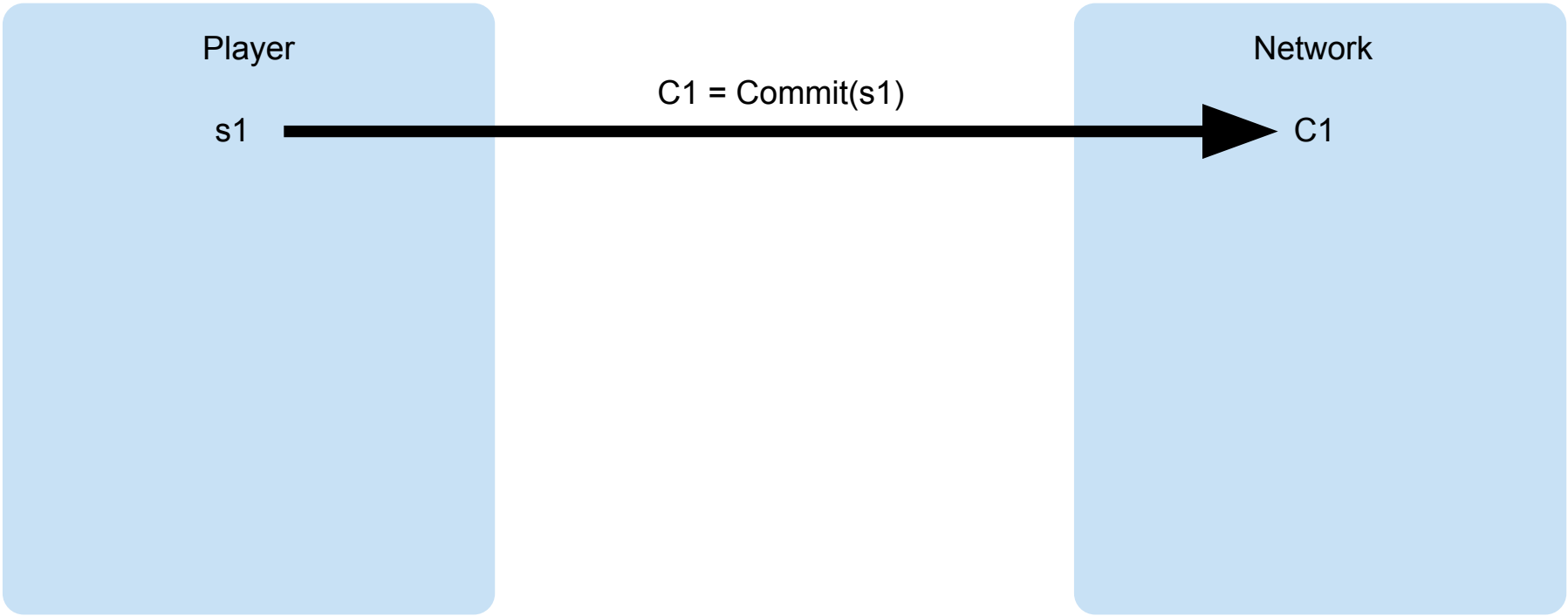

INFORMATION ASYMMETRY
Nagamanen 0.3 < Isoma < The Citadel
Oto
Gallente Federation Contested
Route
No Destination
Agent Missions
the consequences.
Warp scramble attempt from Andy Chieve [ALWAR](Atron) to Alice Rockfist [.VERA] (Phoenix)
Searly1981 Searle [RDRAW] Ninazu 35 km
Lee Church [RDRAW] Dominix 9,411 m
Overview
Distance Name Type Tag Velocity
Selected Item
Searly1981 Searle
Distance: 35 km
Sec: -6.7
Probe Scanner
ANALYZE
Probes
No Probes Deployed
Scan Results
Distance ID Name Group Signal
Fleet (41 Members) / Wing 1 / Squad 4
My Fleet History Fleet Finder
Broadcast History
Clear History
02:02:19 - Solidus Yanumano needs armor
02:02:18 - Evesham needs armor
02:02:08 - Robert Dalentis needs armor
02:02:06 - Target Searly1981 Searle (Ninazu)
02:01:30 - Target Nicole-Reine Lepaute (Apostle)
02:01:14 - Bodewhin Schwarz needs capacitor
02:01:14 - Solidus Yanumano needs armor
02:01:00 - Target Lee Church (Dominix)
Solidus Yanumano needs armor
(No Voice Messages)
* Vlad Cettes throws a hand grenade back
Cycle green > first kill in my carrier :D
Cycle green > Kill: Damian Leon (Armageddon)
Daft-Cube > rip battleship
* Cait Vadam dives at DK
* Diana Kim screams!!
Cycle green > just killed a fax too.
Michael Elieser
Morwen Lagann
Nevyn Auscent
Niraia
Norrin Ellis
Pix Severus
Saede Riordan
Sara Adoulin
Silver Night
Spc One
Thal Vadam
Thelzor Incorentis
Viriel
Vlad Cettes
Zirio
Zombie Berlioz
In Space 959 m/s
In Space 1111 m/s
In Space 1069 m/s

INFORMATION ASYMMETRY

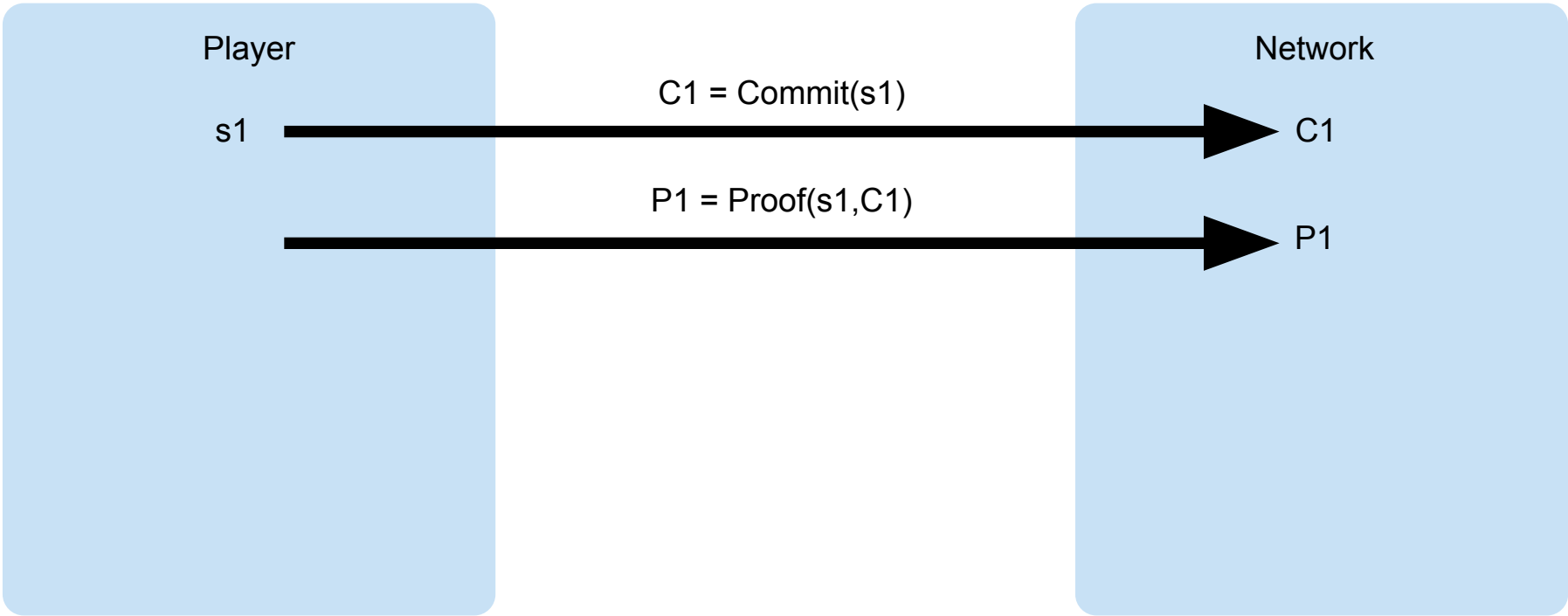
Hidden information on the blockchain



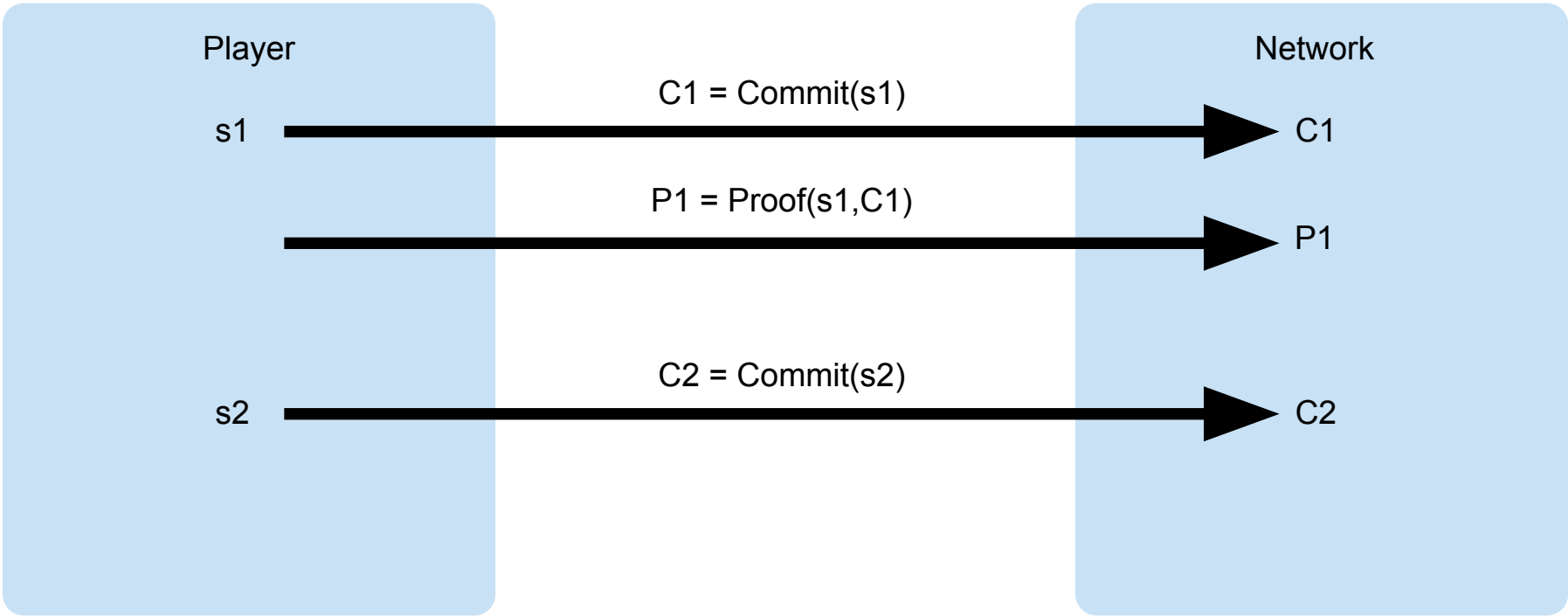
Hidden information on the blockchain



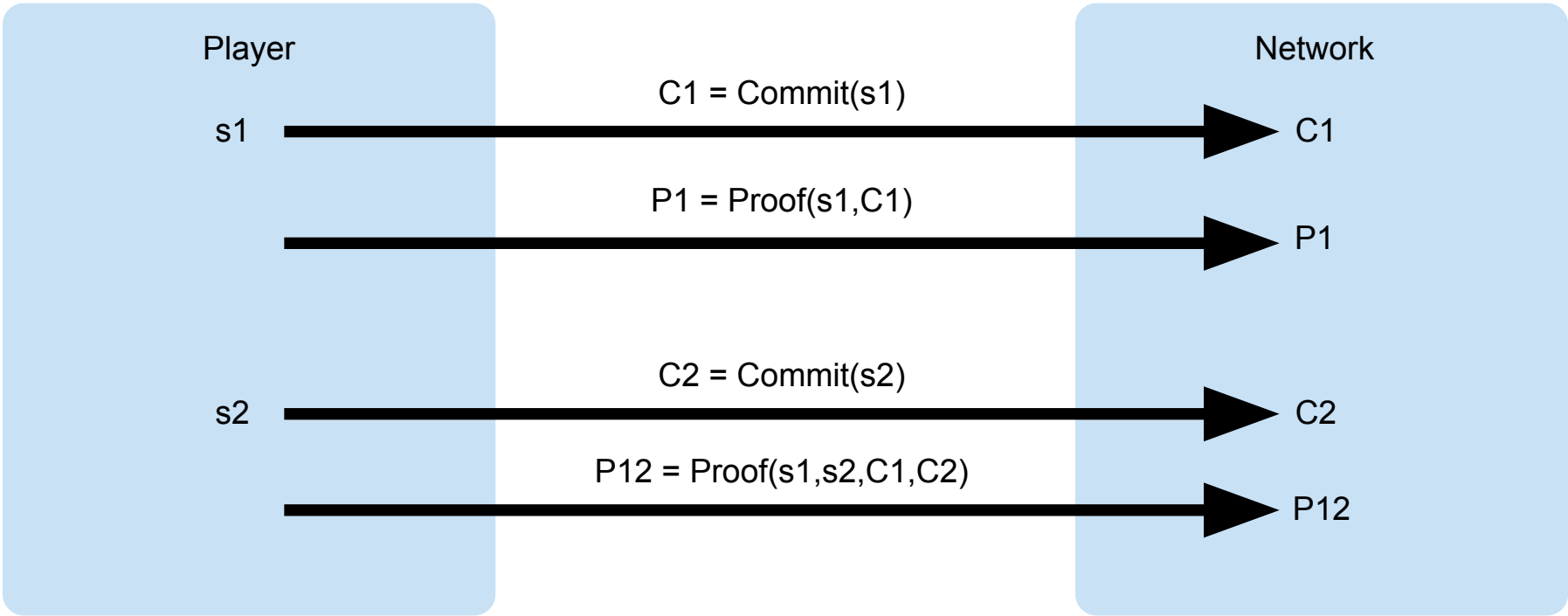
Hidden information on the blockchain



Hidden information on the blockchain



Hidden information on the blockchain



zkSNARKs

I’m drawing a card from a deck and adding it to my hand.

I’m not going to show you my hand, but I can prove that I drew the card at random from a properly-shuffled deck.

Zero-Knowledge Fog of War

I’m moving my knight from secret location A to secret location B.

I’m not going to tell you where A and B are, but I can prove that they are an L-shape away from each other.

Game Construction

- Everyone lives on large 2-D grid

Game Construction

- Everyone lives on large 2-D grid
- For location (x, y), hash(x,y) is location's **public address**.
  - The coordinates themselves are the location's **private address**.

Game Construction

- Everyone lives on large 2-D grid
- For location (x, y), hash(x,y) is location's **public address**.
- Locations (x,y) such that

$$\text{hash}(x,y) < \text{DIFFICULTY\_THRESHOLD}$$

have **habitable planets**. All other spaces are empty.

Game Construction

- Everyone lives on large 2-D grid
- For location (x, y), hash(x,y) is location's **public address**.
- Locations (x,y) such that

  $$\text{hash}(x,y) < \text{DIFFICULTY\_THRESHOLD}$$

  have **habitable planets**. All other spaces are empty.

- Player-controlled **units** live on planets the player owns.

Game Construction: State

Public State

- Which public addresses are owned, who owns them, and their populations

Game Construction: State

Public State

- Which public addresses are owned, who owns them, and their populations

Private State

- Private addresses (x,y) of player’s planets
- Information learned from computation

Player Action: Init

function initializePlayer(uint planetId, uint claimedDist, Proof zkProof)

Initialize a player at coordinates with ID **planetId**. Also checks that these coordinates are within some claimed distance from the origin.

Player Action: Init

function initializePlayer(uint planetId, uint claimedDist, Proof zkProof)

zkProof: I know some coordinates (x, y) such that

- hash(x, y) = planetId
- x^2 + y^2 < claimedDist^2

Player Action: Move

function move(uint fromPlanetId, uint toPlanetId, uint worldRadius, uint maxDist)

Move forces from **fromPlanetId** to **toPlanetId**.

- Check that both planets are “in bounds”
- Pay some cost depending on **maxDist** between the two planets.

Player Action: Move

function move(uint fromPlanetId, uint toPlanetId, uint worldRadius, uint maxDist)

zkProof: I know some coordinates (x1, y1) and (x2, y2) such that

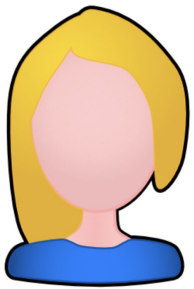
- hash(x1, y1) = fromPlanetId
- hash(x2, y2) = toPlanetId
- x2^2 + y2^2 < worldRadius^2
- (x1-x2)^2 + (y1-y2)^2 < distMax^2

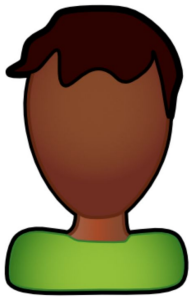
Demo

Agenda

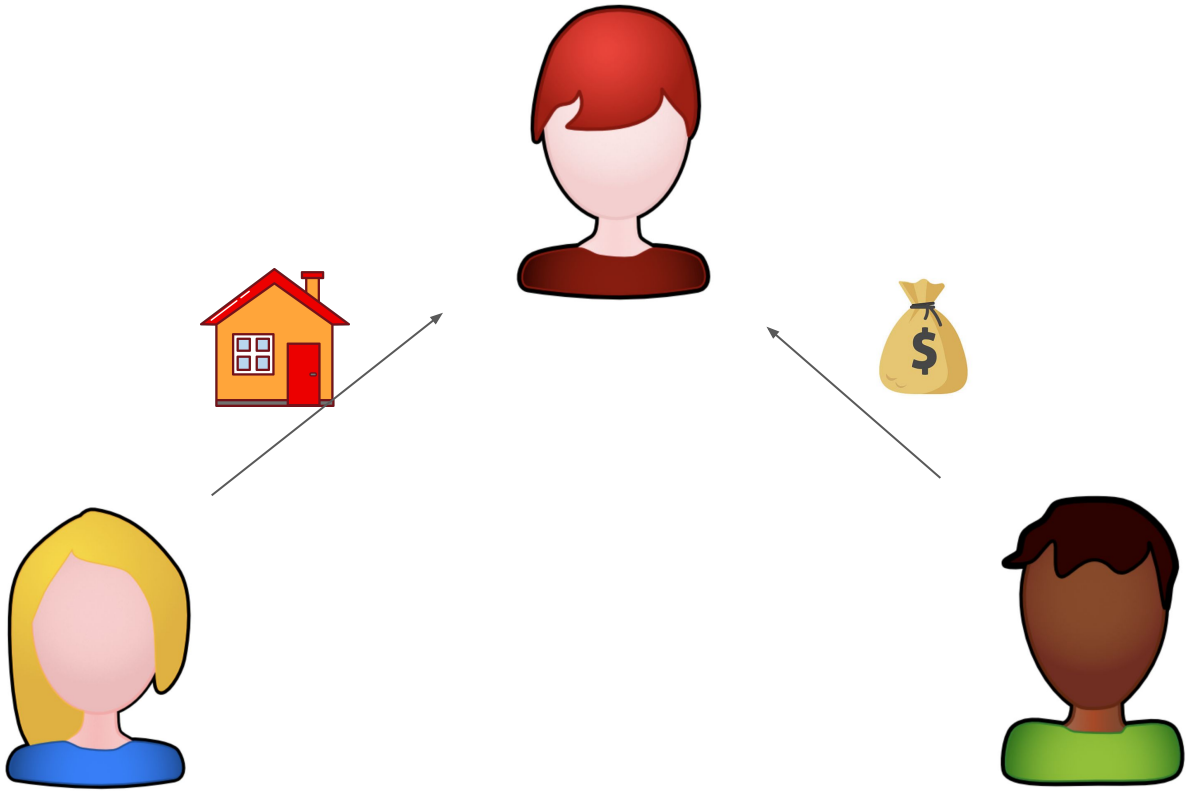
- Preamble: zkSNARKs are “programmable” cryptography
- Part 1: How to use zkSNARKs in apps
  - Group / ring signatures
  - Anonymous polling / private cryptocurrency
  - Dark Forest
  - **Information marketplaces**
- Part 2: Three categories of zkSNARK applications
  - Adding hidden information to decentralized systems
  - Speeding up blockchains
  - Language of Truth

Escrow

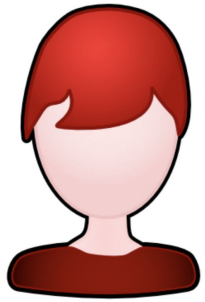


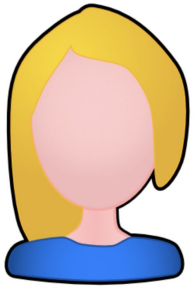


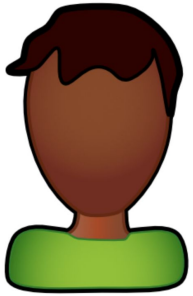
Escrow



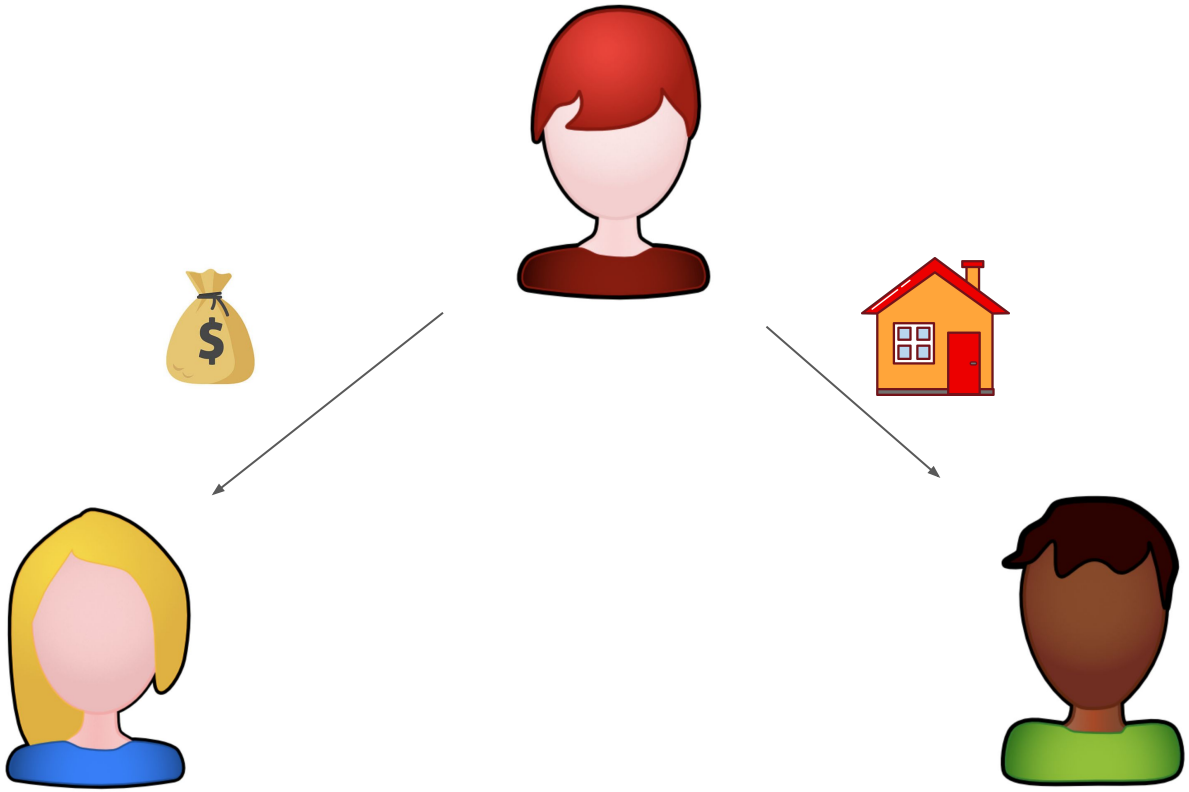
Escrow







Escrow



Escrow


The New York Times

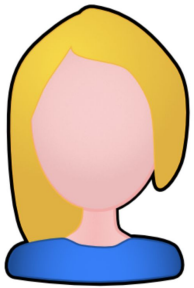
LATE CITY EDITION

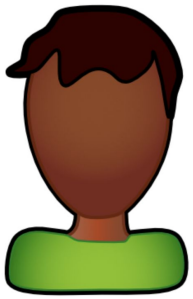
MEN WALK ON MOON

ASTRONAUTS LAND ON A PLAIN
AFTER STEERING PAST CRATER

Voice From Moon:
'Eagle Has Landed'

A Powdery Surface
Found by Armstrong

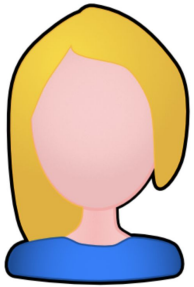


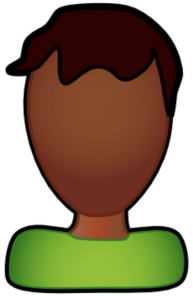


Escrow


The New York Times
LATE CITY EDITION
MEN WALK ON MOON
ASTRONAUTS LAND ON A PLAIN
AFTER STEERING PAST CRATER
Voice From Moon:
'Eagle Has Landed'
A Powdery Surface
Found by Armstrong


$





Escrow

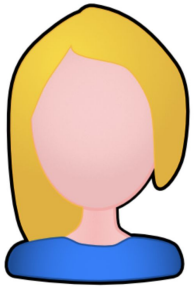

The New York Times

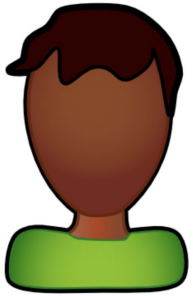
MEN WALK ON MOON

ASTRONAUTS LAND ON A PLAIN AFTER STEERING PAST CRATER

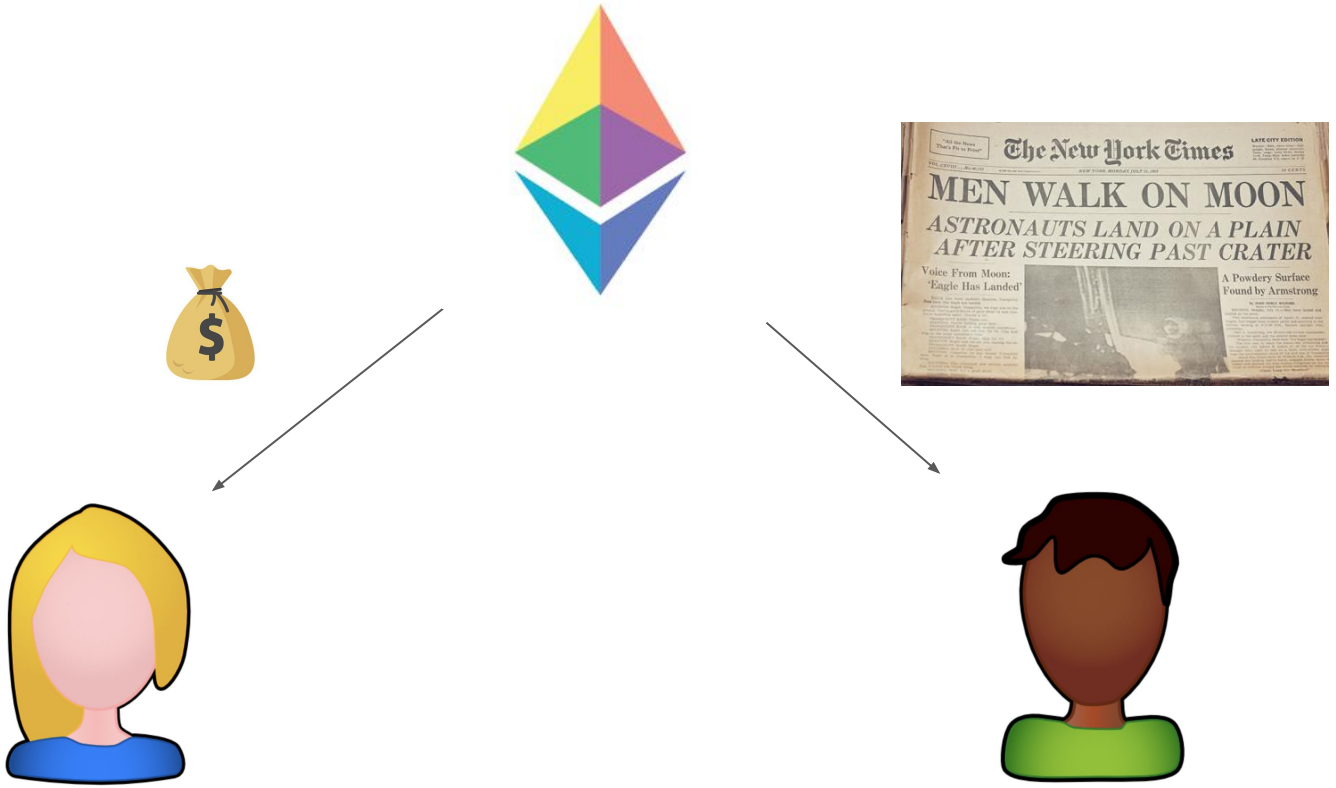
Voice From Moon: 'Eagle Has Landed'

A Powdery Surface Found by Armstrong

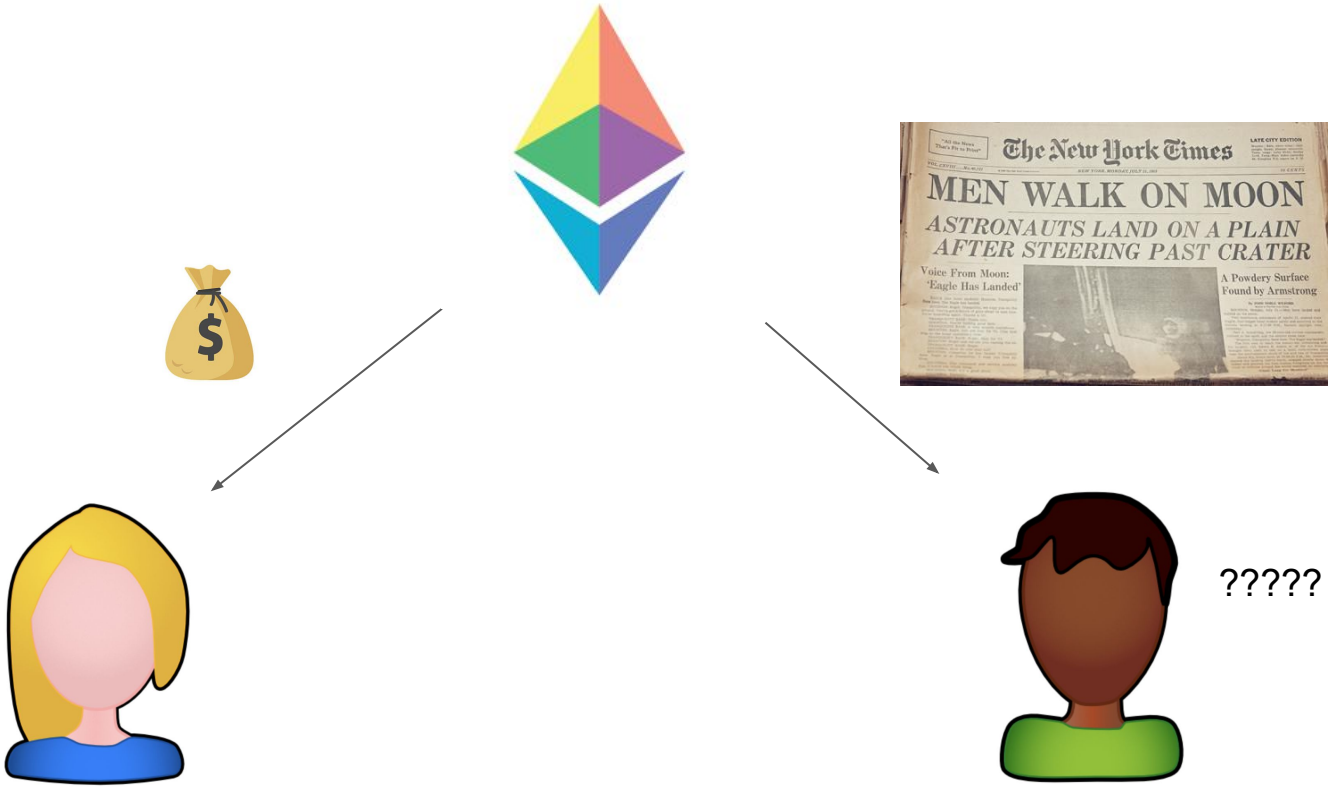




Escrow


The New York Times
MEN WALK ON MOON
ASTRONAUTS LAND ON A PLAIN
AFTER STEERING PAST CRATER
Voice From Moon:
'Eagle Has Landed'
A Powdery Surface
Found by Armstrong

Escrow


The New York Times
MEN WALK ON MOON
ASTRONAUTS LAND ON A PLAIN
AFTER STEERING PAST CRATER
Voice From Moon:
'Eagle Has Landed'
A Powdery Surface
Found by Armstrong
?????

On-Chain Marketplace

- Toy example: Bob wants to buy the preimage of 0x98b3f001 from Alice

On-Chain Marketplace

Escrow contract checks that both buyer and seller have fulfilled conditions:

- Bob has locked up $$ in escrow
- Alice has published the data the buyer wants

Problem:

- The only way that the contract can check seller's condition is if it can read the seller's data!

Solution

- Alice publishes data encrypted with buyer's public key
- Alice also publishes a zkSNARK proof that this ciphertext is the correct data, encrypted with Bob's public key
- Smart contract only releases funds to Alice if zkSNARK proof verifies

https://github.com/nulven/EthDataMarketplace - Nick Ulven (2021)

Solution

Public Input:

- Buyer public key **pk**
- Ciphertext **c**
- Commitment **h**

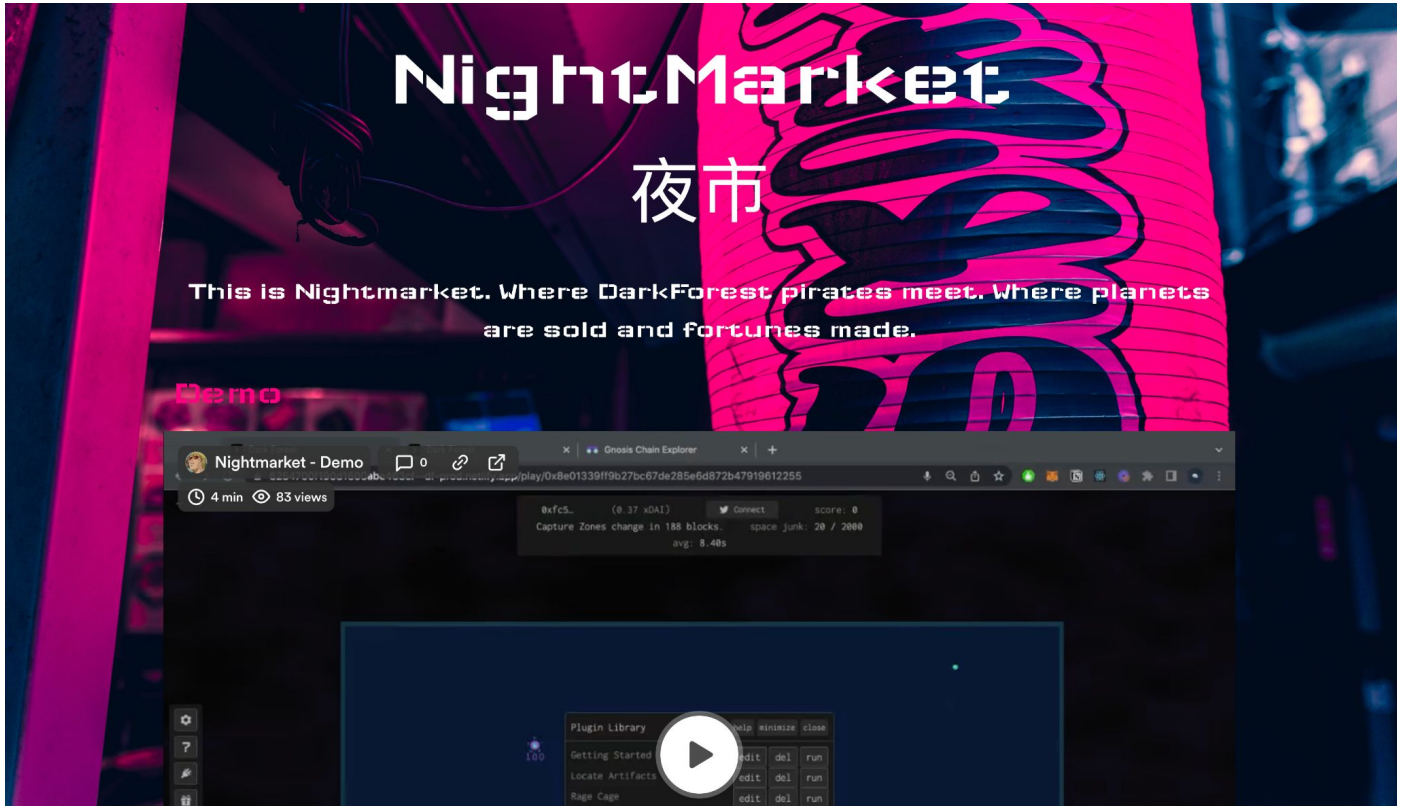
Private Input:

- Secret data **s**

Proves:

- Hash(s) = h (or property(s) = h)
- Enc(s, pk) = c

Nightmarket



Nightmarket

https://blog.zkga.me/nightmarket - 0xSage, xyz_pierre (2022)

https://nightmart.xyz/

Nightmarket

Key constraints are as follows:

Seller

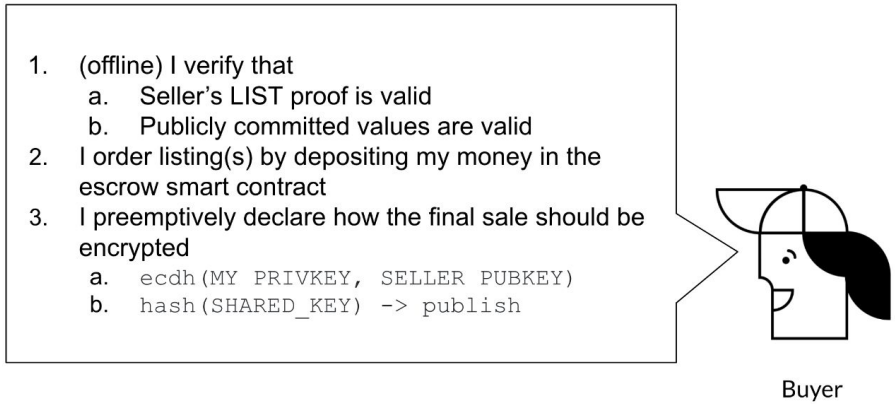
Constraint	Publicly Verifiable Value
`hash(PLANET_X/Y,PLANETHASH_KEY)`	A valid planet hash
`perlin(PLANET_X/Y,BIOMEBASE_KEY)`	The correct biomebase
`poseidon_encode_check(CIPHERTEXT, PLANET_ADDR, KEY)`	Valid ciphertext using KEY
`hash(KEY)`	KEY won't change later
`SELLER_ADDR * SELLER_ADDR`	Watermark proof to Seller

In detail, we ensure:

- `hash(PLANET_X, PLANET_Y, PLANETHASH_KEY)`: Seller proves that they know a planet coordinate, which buyers can later verify on-chain.
- `poseidon_encode_check(CIPHERTEXT, PLANET_X, PLANET_Y, KEY)`: Seller commits to having correctly encrypted a `CIPHERTEXT`, using a symmetric `KEY` on the planet coordinates. The seller does publish the cipher on-chain for later decryption. For the Poseidon encryption check itself, we adapted @weijie's Poseidon circuits to constrain correct encoding.
- `hash(KEY)`: Seller separately commits to the secret `KEY` that was used for the above step. Notably, the actual item being sold isn't the raw planet coordinate, but this secret `KEY` used to symmetrically encrypt the coordinates. Possession of the `KEY` means that anyone can subsequently decrypt the ciphertext and retrieve the original planet coordinates.

Nightmarket

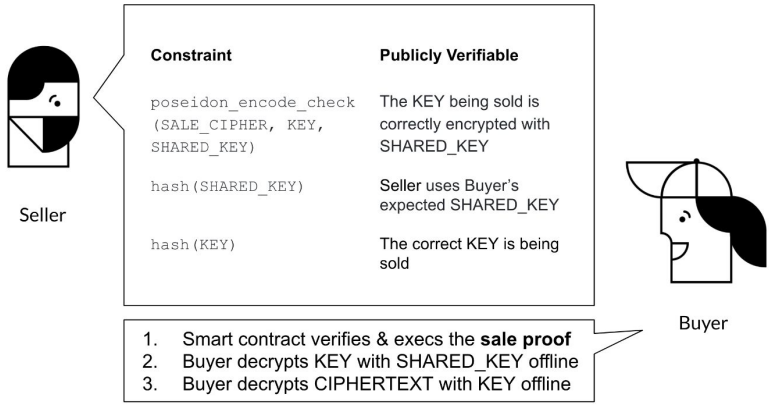
Then multiple buyers can make purchase orders on a single listing, as follows:



In detail, Buyers signal their interest by depositing some amount into the escrow contract, along with an expectation of a `SHARED_KEY` that the seller later uses to symmetrically encrypt the `KEY` being sold. The Buyer constructs this `SHARED_KEY` offline, through an ECDH key exchange scheme. She publishes a hash of this `SHARED_KEY` on-chain, along with her deposit. More on this in step 3.

Nightmarket

Finally, the Seller fulfills Buyer's purchase orders with a Sale circuit, proving the following:



Specifically, the Seller also performs an ECDH key exchange offline, generating the same `SHARED_KEY` as Buyer. Seller uses the `SHARED_KEY` to encrypt the original `KEY` and broadcasts the encrypted `SALE_CIPHER` on-chain.

Later, the Buyer can privately decrypt this `SALE_CIPHER` with the `SHARED_KEY`, to obtain `KEY`. Then, the Buyer can decrypt `CIPHERTEXT` with the `KEY`, and retrieve the original planet coordinates.

What else can we sell?

- A Bitcoin, Ethereum, SSH, DKIM private key
- A smart contract exploit (or program exploit more generally)
- A picture of a bird
- …

Agenda

- Preamble: zkSNARKs are “programmable” cryptography
- Part 1: How to use zkSNARKs in apps
  - Group / ring signatures
  - Anonymous polling / private cryptocurrency
  - Dark Forest
  - Information marketplaces
- **Part 2: Three categories of zkSNARK applications**
  - Adding hidden information to decentralized systems
  - Speeding up blockchains
  - Language of Truth

Agenda

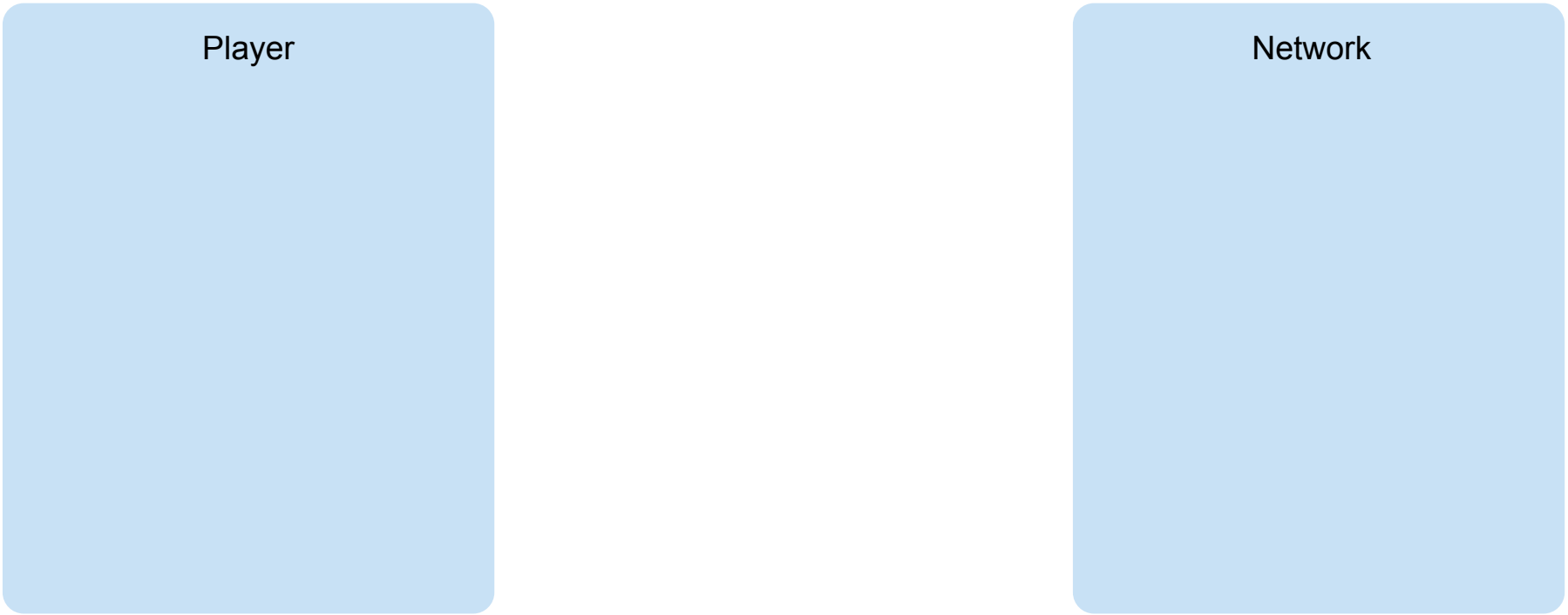
- Preamble: zkSNARKs are “programmable” cryptography
- Part 1: How to use zkSNARKs in apps
  - Group / ring signatures
  - Anonymous polling / private cryptocurrency
  - Dark Forest
  - Information marketplaces
- Part 2: Three categories of zkSNARK applications
  - **Adding hidden information to decentralized systems**
  - Speeding up blockchains
  - Language of Truth

Pattern #1: privacy on decentralized systems

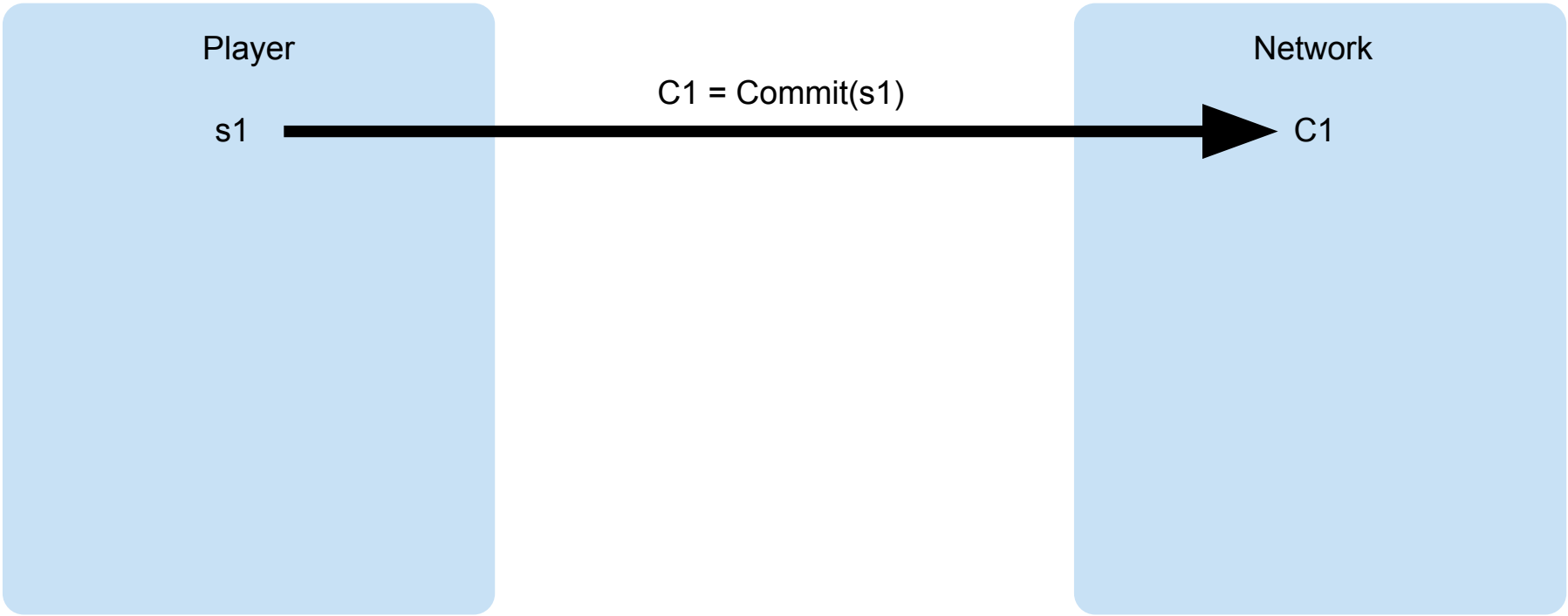
Privacy on decentralized systems

- Sensitive social, financial, or professional data
  - e.g. private bids in an auction
- My ownership of a crypto token
  - e.g. private cryptocurrency
- A private hand of cards in a game
  - e.g. Dark Forest

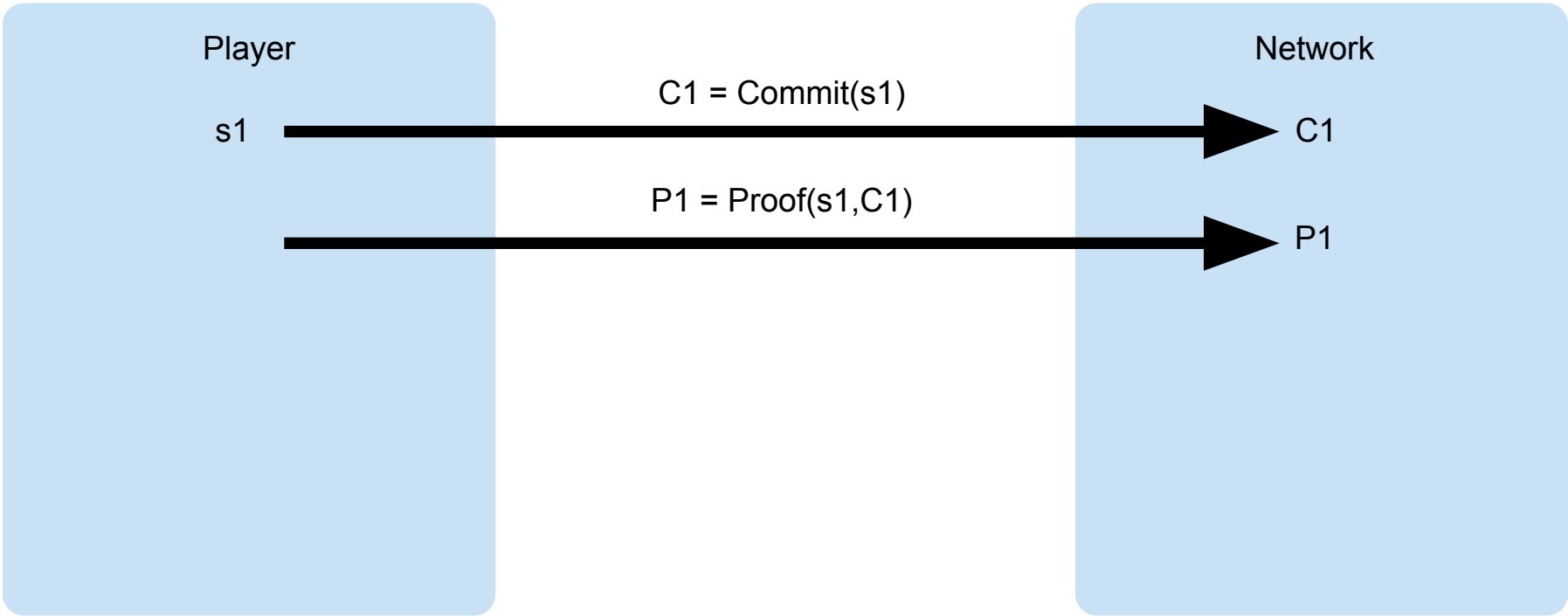
Hidden information on the blockchain



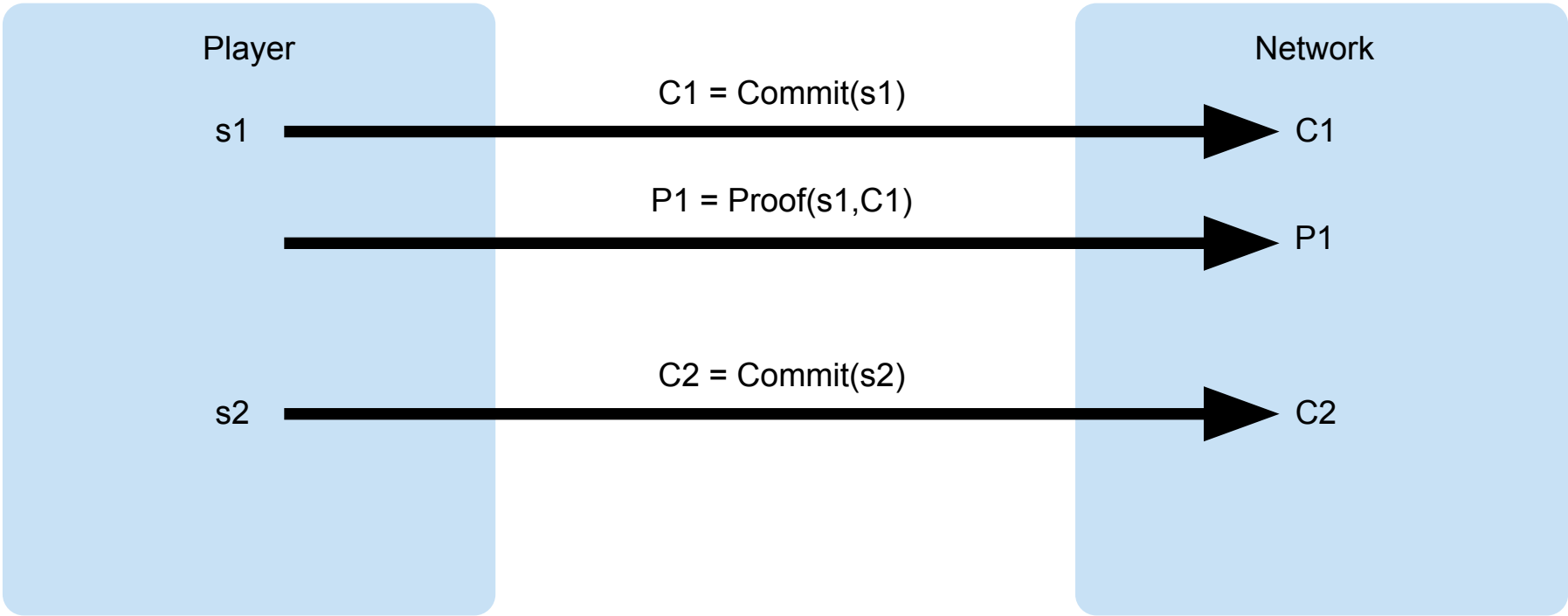
Hidden information on the blockchain



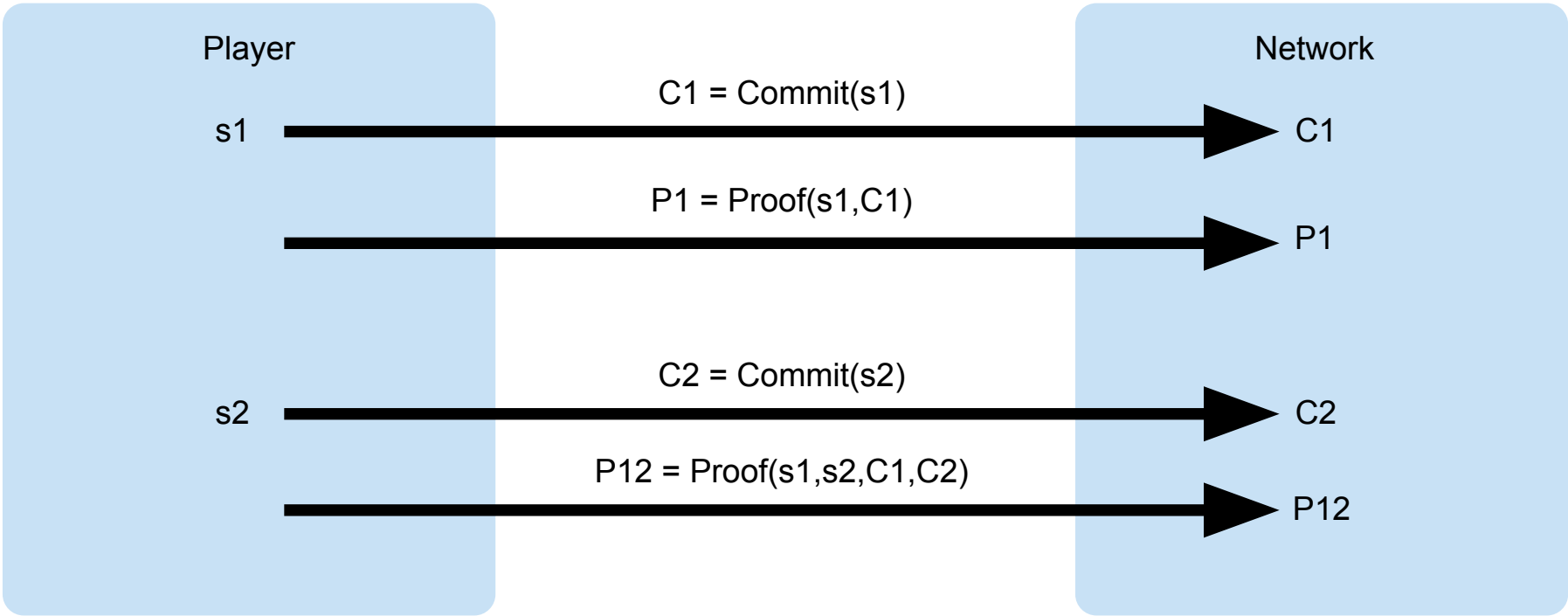
Hidden information on the blockchain



Hidden information on the blockchain



Hidden information on the blockchain



Privacy on decentralized systems

Privacy on decentralized systems

Privacy on decentralized systems

Privacy on decentralized systems

Privacy on decentralized systems

Social, professional, medical, scient
etc.?

Agenda

- Preamble: zkSNARKs are “programmable” cryptography
- Part 1: How to use zkSNARKs in apps
  - Group / ring signatures
  - Anonymous polling / private cryptocurrency
  - Dark Forest
  - Information marketplaces
- Part 2: Three categories of zkSNARK applications
  - Adding hidden information to decentralized systems
  - **Speeding up blockchains**
  - Language of Truth

Pattern #2: accelerating blockchains

Blockchains

Programmable blockchains are decentralized networks for running computations.

The security of a blockchain comes from the fact that everyone on the network runs and checks all the computations!

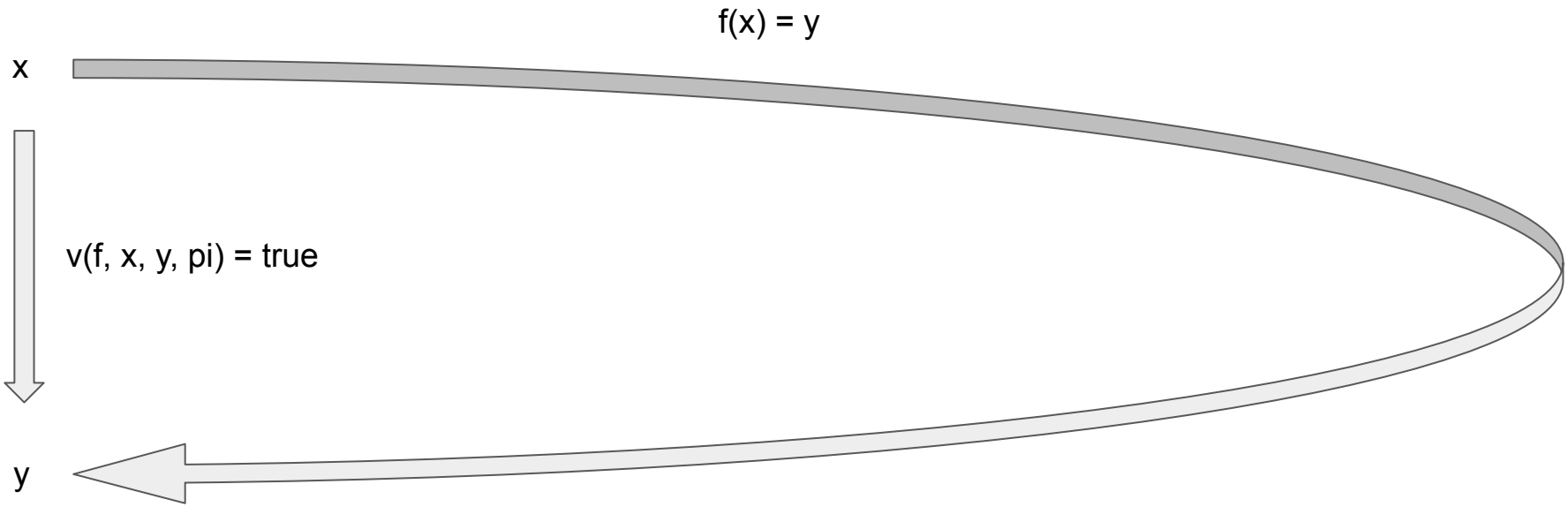
Blockchains

Programmable blockchains are decentralized networks for running computations.

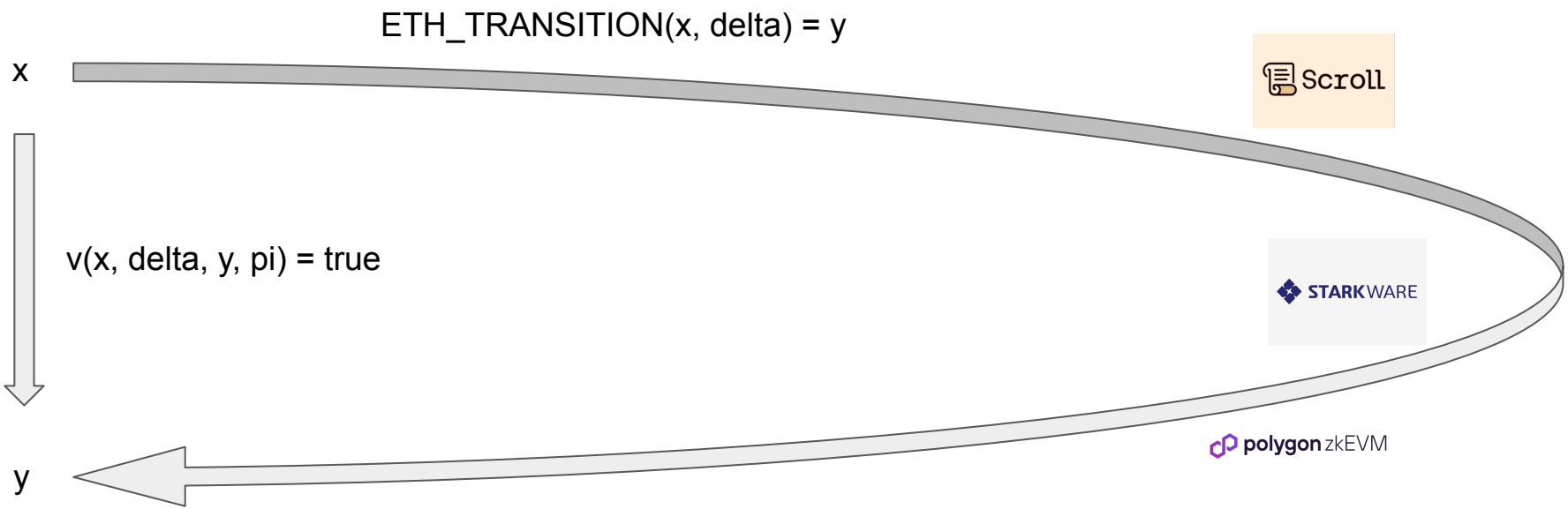
The security of a blockchain comes from the fact that everyone on the network runs and checks all the computations!

This is also where we run into performance bottlenecks…

zkSNARKs are like “computational wormholes”



zkSNARKs are like “computational wormholes”



Co-processors: Verifiable Computation

What other specific (expensive) computations might we want a decentralized network to run?

Co-processors: Verifiable Computation and ZKML

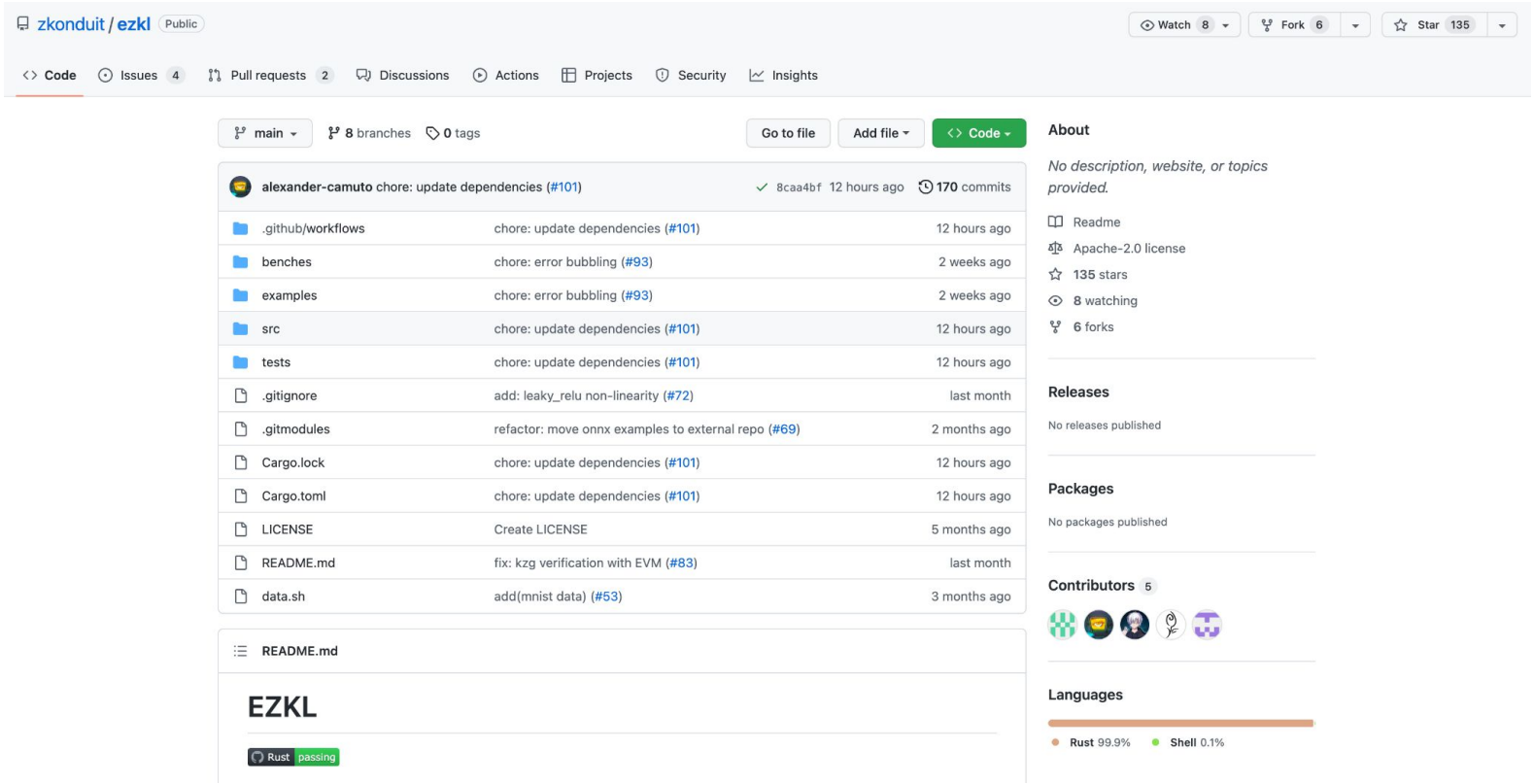
In the future, suppose a neural network or LLM is run to make a judgement on-chain.

- Ex: running a financial strategy, attesting to the sentiment of today's news

Running a neural net is expensive! Who runs the LLM? Are they running it correctly? What if the LLM's model parameters are proprietary?

zkSNARKs enable one person on the network to run the computation, and everyone to get (and trust) the result.

Co-processors: Verifiable Computation and ZKML



Verifiable Computation

Before any caseload: OpenAI commits to the model commit(model) = C

Then, on any run, proves…

Verifiable Computation

Public inputs:

- Input **x**
- Claimed output **y**
- Model commitment **c**

Private inputs:

- Model **M**

Proves:

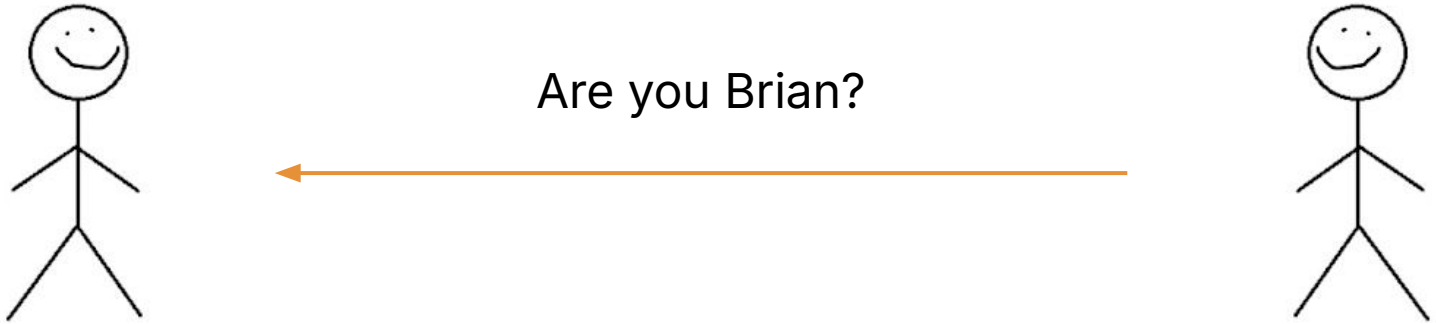
- **M**(**x**) = **y**
- commit(**M**) = **c**

Agenda

- Preamble: zkSNARKs are “programmable” cryptography
- Part 1: How to use zkSNARKs in apps
  - Group / ring signatures
  - Anonymous polling / private cryptocurrency
  - Dark Forest
  - Information marketplaces
- Part 2: Three categories of zkSNARK applications
  - Adding hidden information to decentralized systems
  - Speeding up blockchains
  - **Language of Truth**

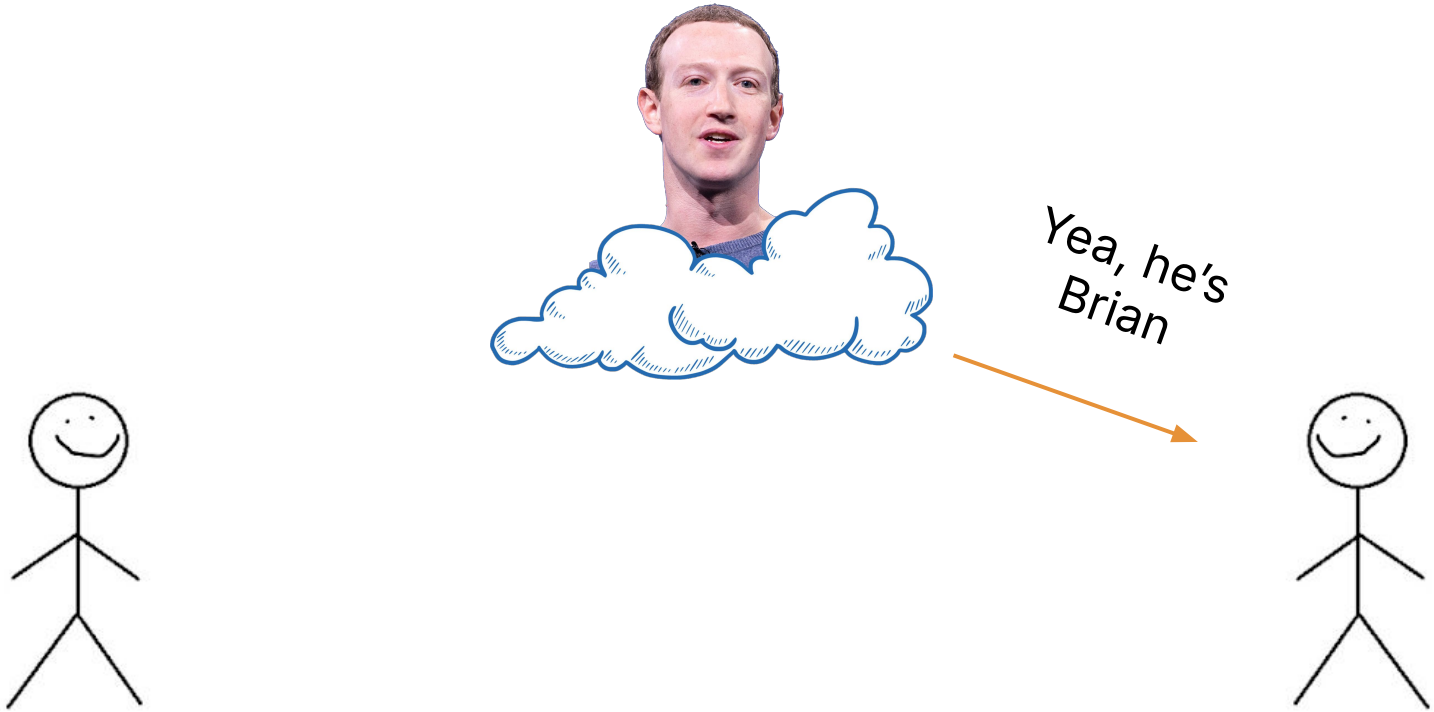
General problem: someone on the Internet wants to ask someone on the Internet for some data

Digital Communication Today

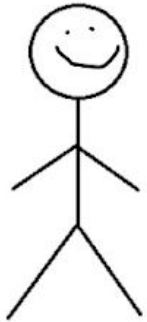


Digital Communication Today

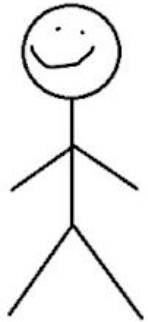
Digital Communication Today



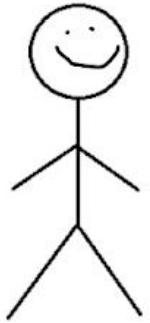
Digital Communication Today



What’s your credit score?

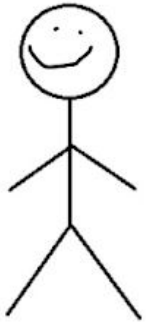


Digital Communication Today



Please respond for me 🥺

EQUIFAX



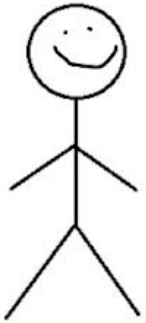
Digital Communication Today

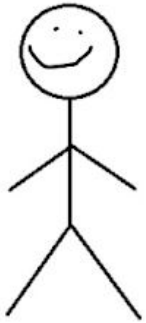
CHASE

HSBC

gathers your data

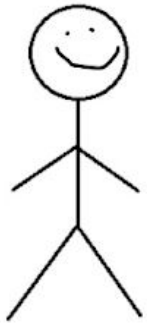
EQUIFAX



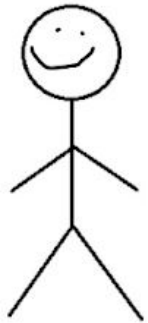


Digital Communication Today

Digital Communication Today

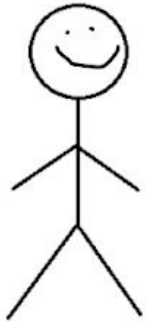


Given an arbitrary function f,
compute for me f(your personal data)

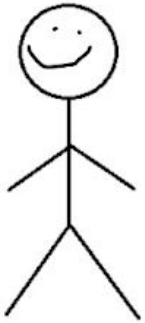


Digital Communication Today

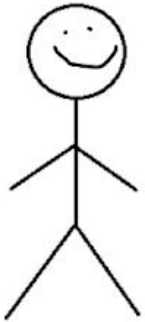
CASE 1



HERE'S WHAT YOU NEED TO KNOW ABOUT BRIAN

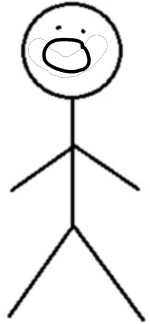


CASE 2

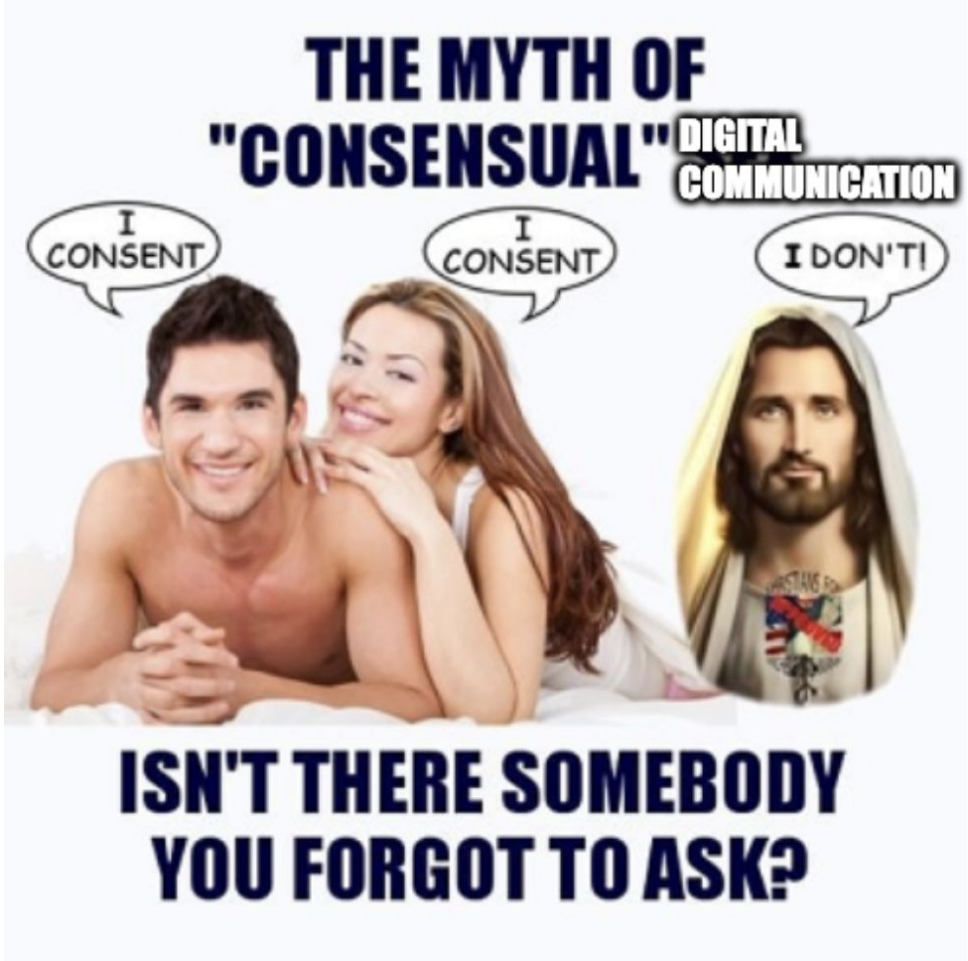


THAT API DOES NOT EXIST

OPERATION NOT ALLOWED

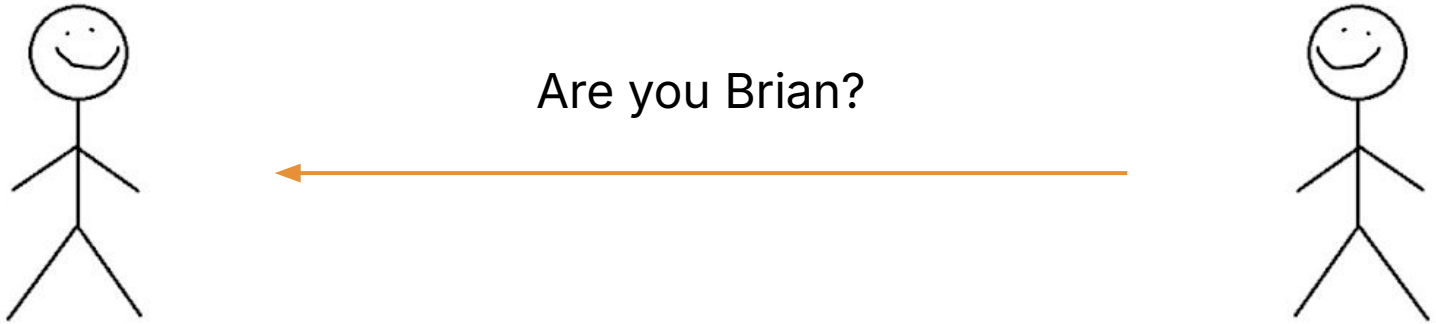


THAT DATA IS SPLIT AMONGST 10 PROVIDERS WHO DON'T HAVE A LICENSING AGREEMENT

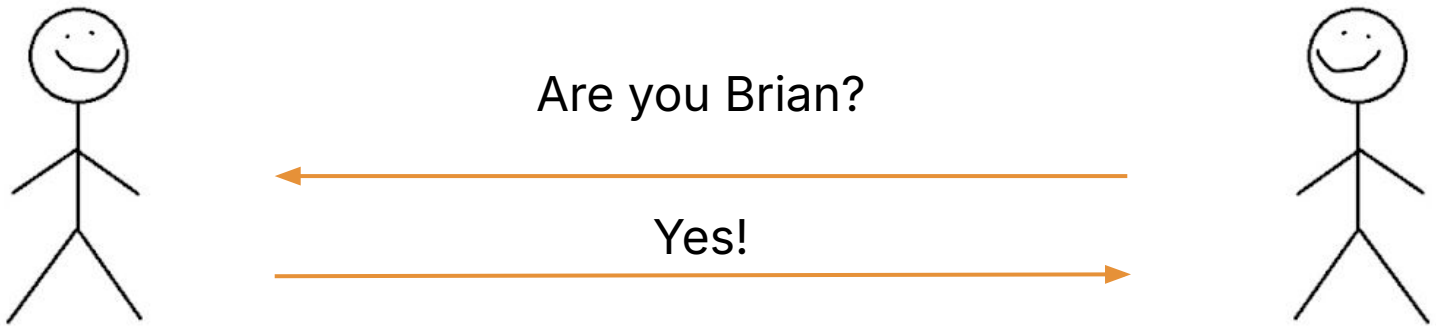

THE MYTH OF
"CONSENSUAL"
DIGITAL
COMMUNICATION
I
CONSENT
I
CONSENT
I DON'T!
ISN'T THERE SOMEBODY
YOU FORGOT TO ASK?


THE MYTH OF
"CONSENSUAL"
DIGITAL
COMMUNICATION
I CONSENT
I CONSENT
I DON'T!
ISN'T THERE SOMEBODY
YOU FORGOT TO ASK?

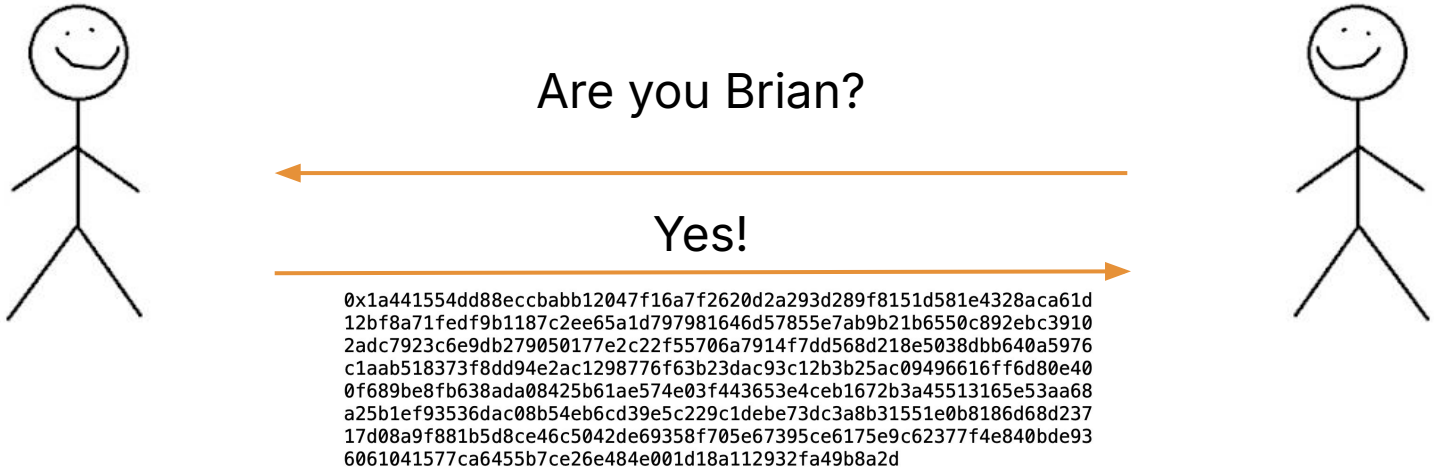
The World with Programmable Cryptography



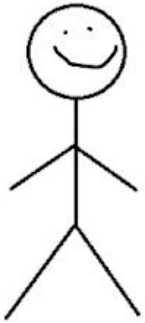
The World with Programmable Cryptography



The World with Programmable Cryptography

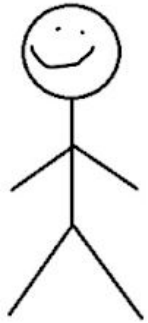


The World with Programmable Cryptography



Are you Brian?

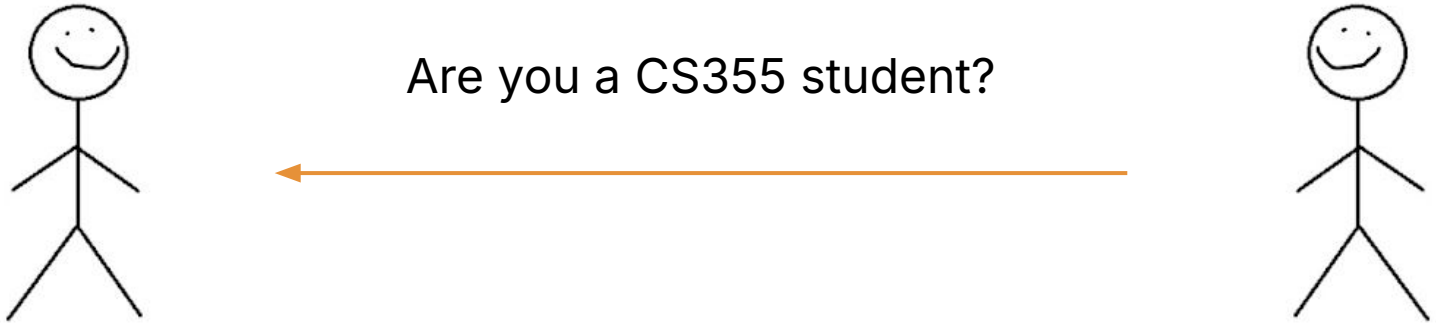
Yes!



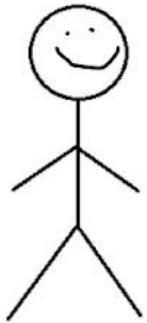
sad Zuck noises


```
0x1a441554dd88eccbabb12047f16a7f2620d2a293d289f8151d581e4328aca61d
12bf8a71fedf9b1187c2ee65a1d797981646d57855e7ab9b21b6550c892ebc3910
2adc7923c6e9db279050177e2c22f55706a7914f7dd568d218e5038dbb640a5976
c1aab518373f8dd94e2ac1298776f63b23dac93c12b3b25ac09496616ff6d80e40
0f689be8fb638ada08425b61ae574e03f443653e4ceb1672b3a45513165e53aa68
a25b1ef93536dac08b54eb6cd39e5c229c1debe73dc3a8b31551e0b8186d68d237
17d08a9f881b5d8ce46c5042de69358f705e67395ce6175e9c62377f4e840bde93
6061041577ca6455b7ce26e484e001d18a112932fa49b8a2d
```

The World with Programmable Cryptography

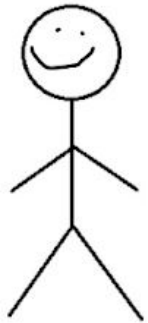


The World with Programmable Cryptography



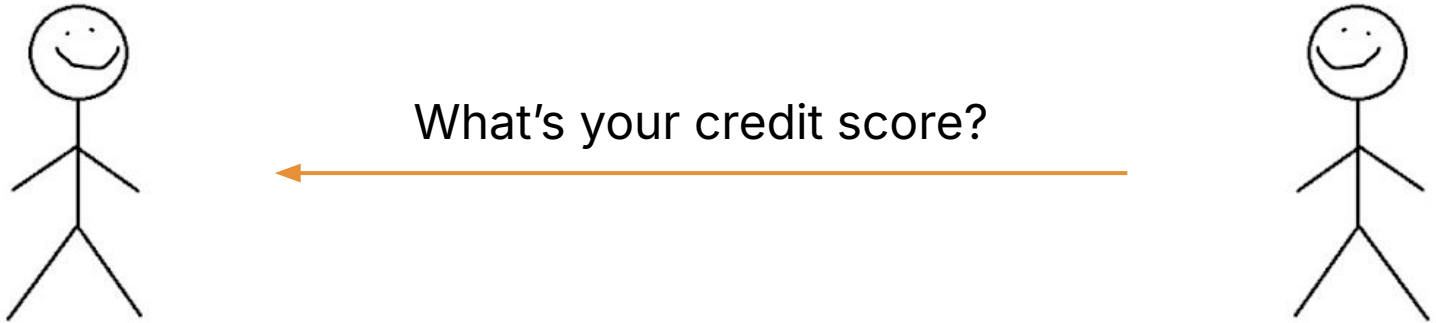
Are you a CS355 student?

Yes!

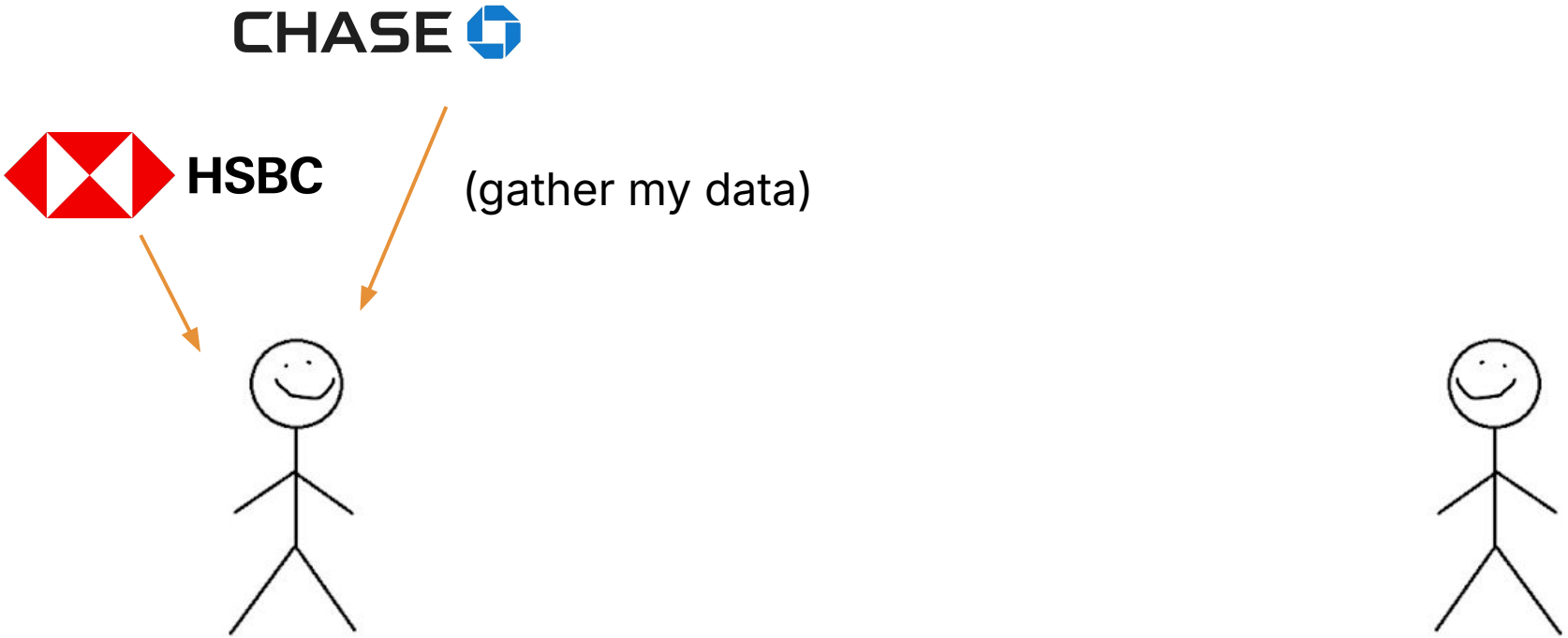


```
0x1a441554dd88eccbabb12047f16a7f2620d2a293d289f8151d581e4328aca61d
12bf8a71fedf9b1187c2ee65a1d797981646d57855e7ab9b21b6550c892ebc3910
2adc7923c6e9db279050177e2c22f55706a7914f7dd568d218e5038dbb640a5976
c1aab518373f8dd94e2ac1298776f63b23dac93c12b3b25ac09496616ff6d80e40
0f689be8fb638ada08425b61ae574e03f443653e4ceb1672b3a45513165e53aa68
a25b1ef93536dac08b54eb6cd39e5c229c1debe73dc3a8b31551e0b8186d68d237
17d08a9f881b5d8ce46c5042de69358f705e67395ce6175e9c62377f4e840bde93
6061041577ca6455b7ce26e484e001d18a112932fa49b8a2d
```

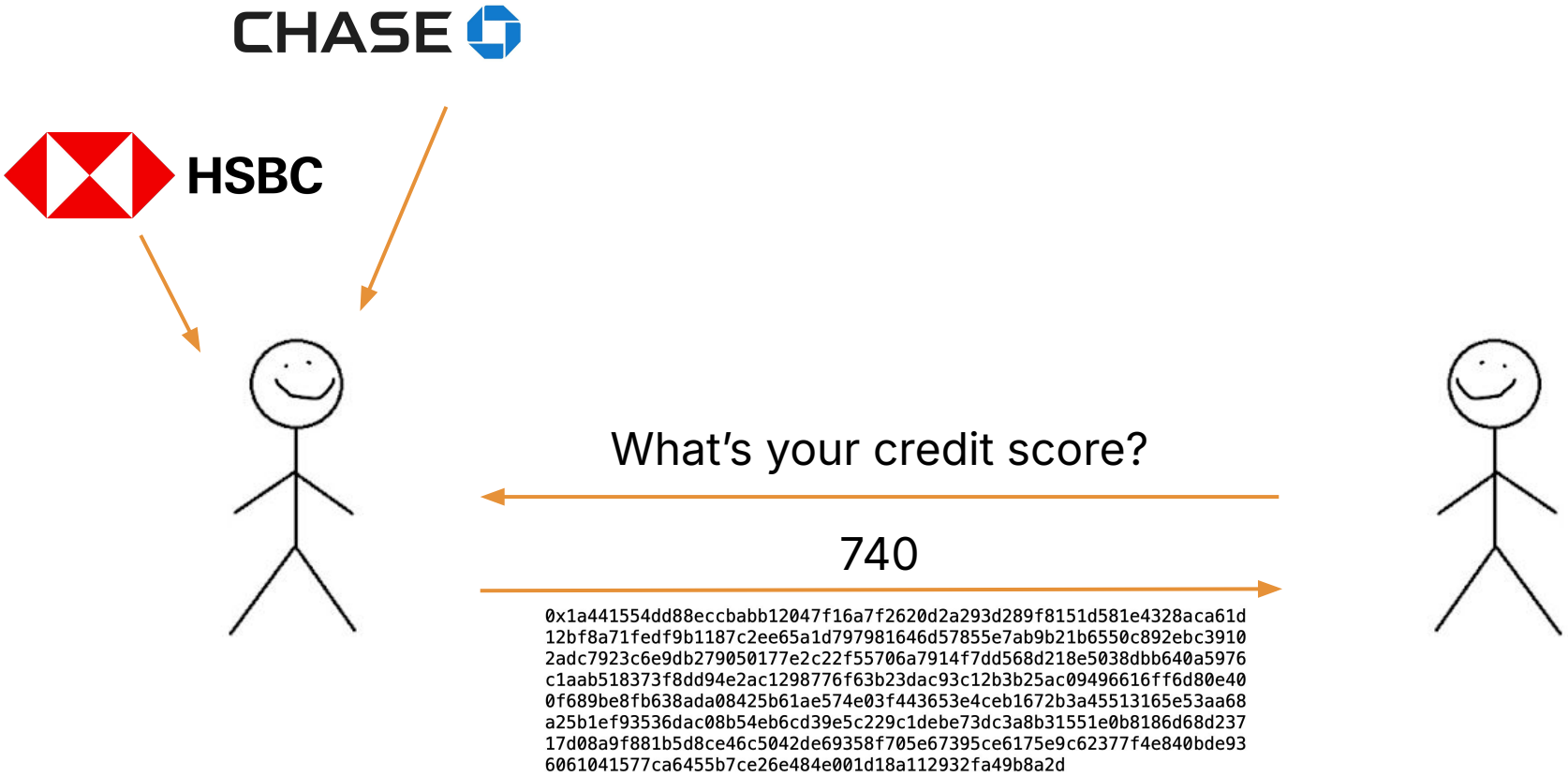
The World with Programmable Cryptography



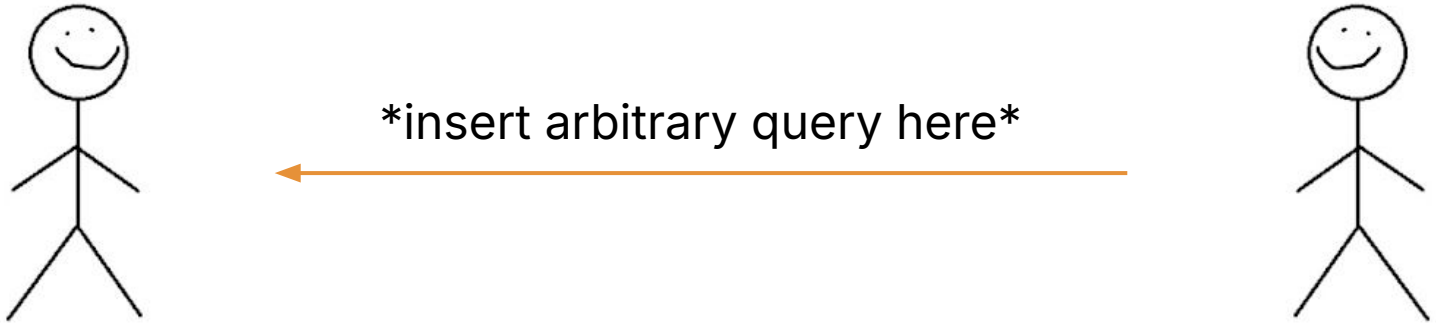
The World with Programmable Cryptography



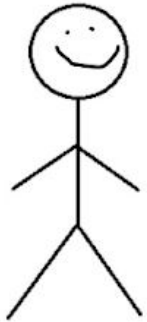
The World with Programmable Cryptography



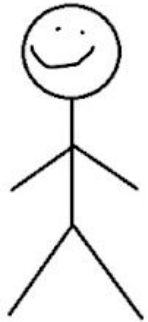
The World with Programmable Cryptography



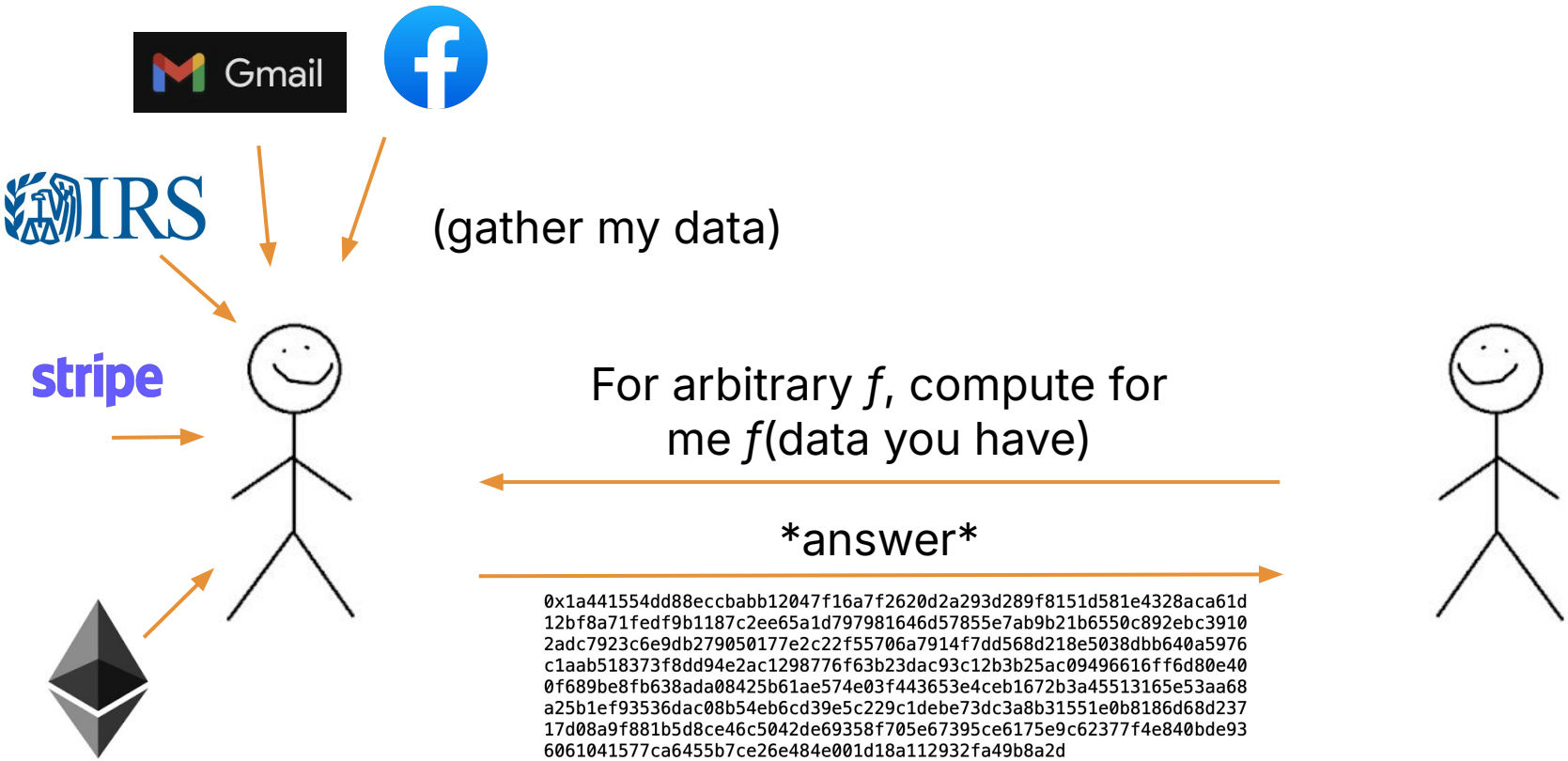
The World with Programmable Cryptography



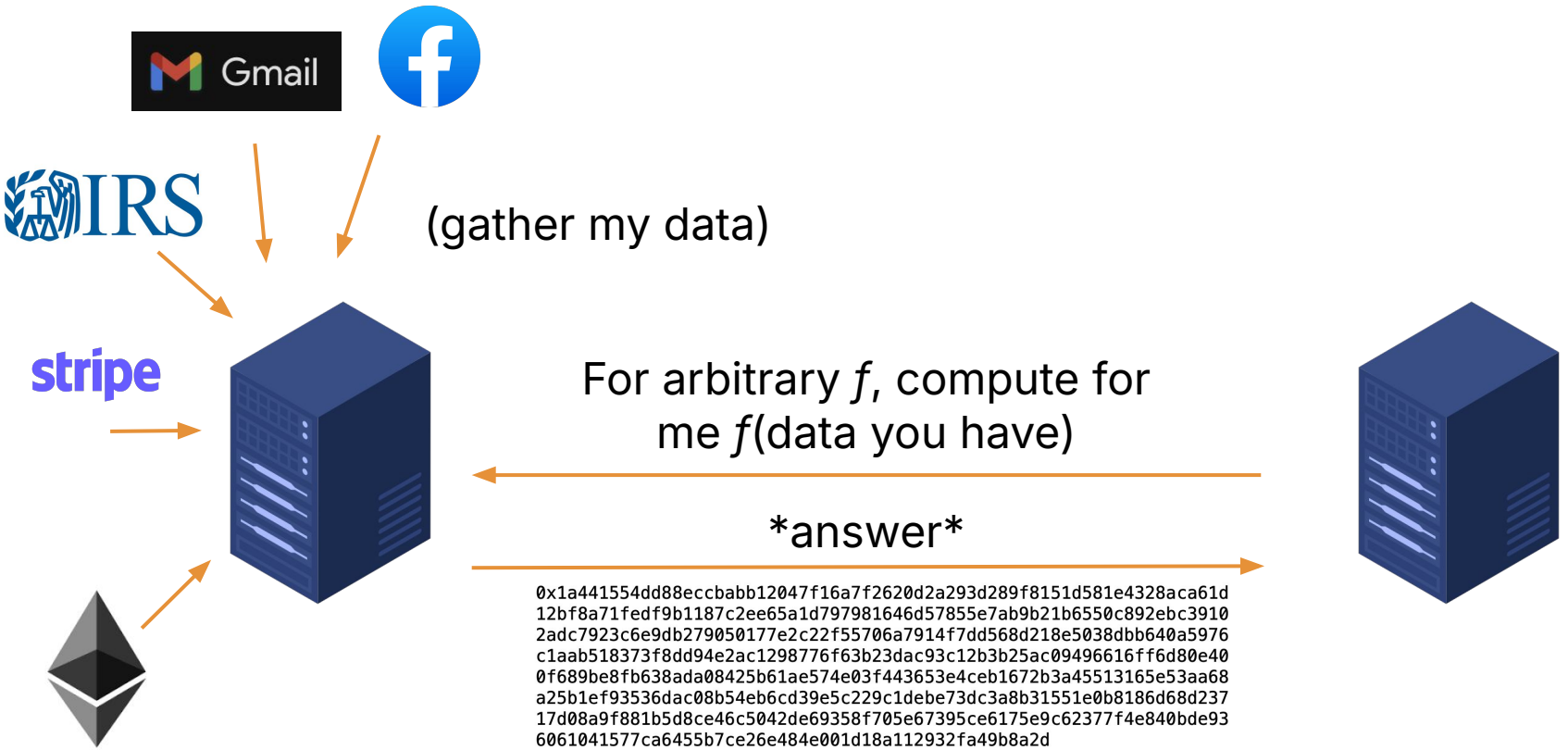
For arbitrary f, compute for me f(data you have)

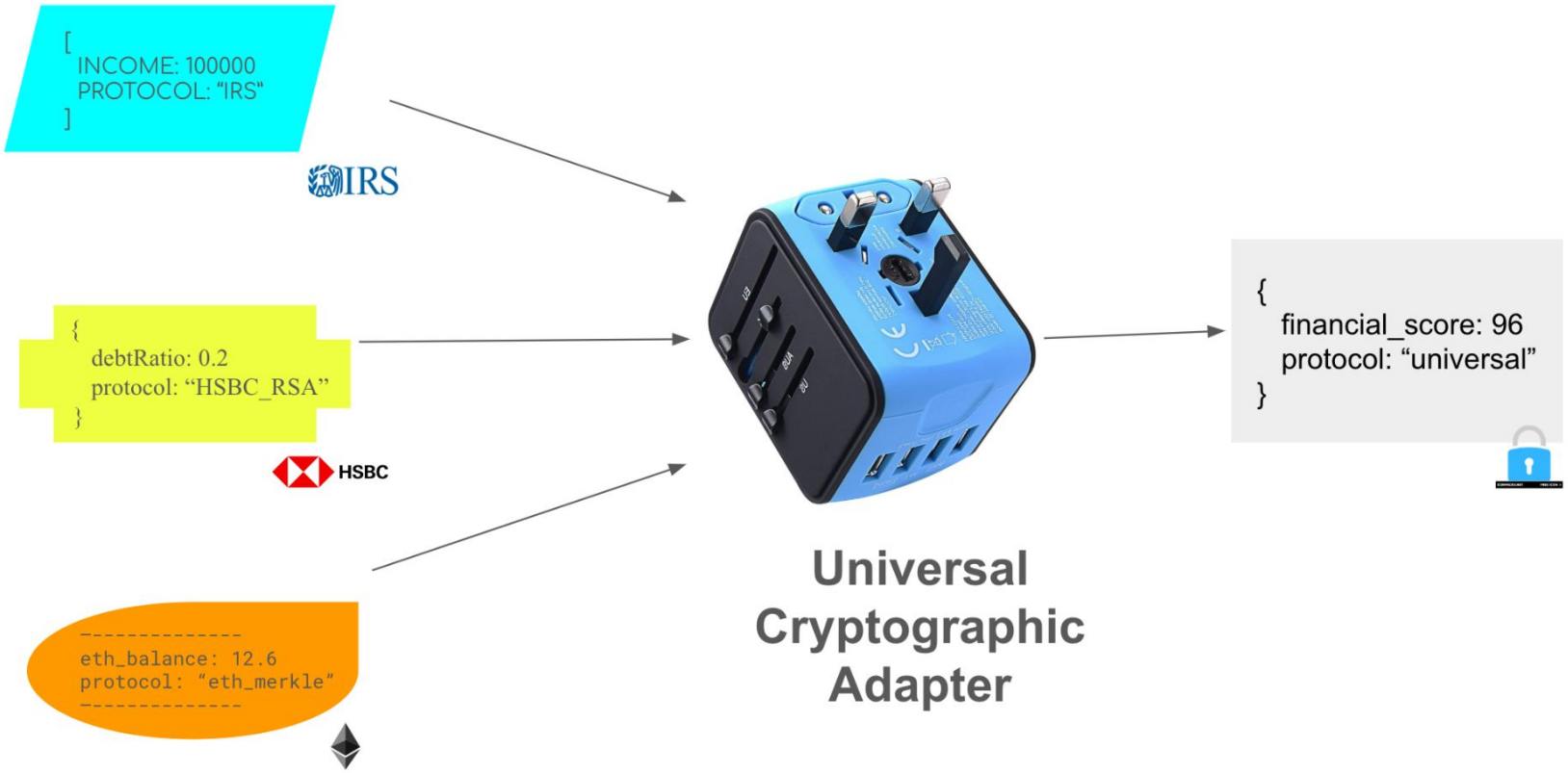


The World with Programmable Cryptography



The World with Programmable Cryptography




[
INCOME: 100000
PROTOCOL: "IRS"
]
IRS
{
debtRatio: 0.2
protocol: "HSBC_RSA"
}
HSBC
---
eth_balance: 12.6
protocol: "eth_merkle"

{
financial_score: 96
protocol: "universal"
}
Universal
Cryptographic
Adapter

Examples

ZK EMAIL

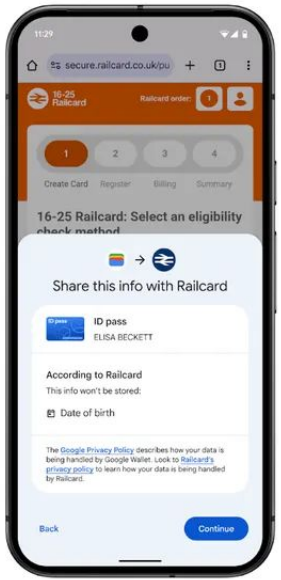
world

Docs

TLSNotary

Proof of data authenticity

Examples



It’s now easier to prove age and identity with Google Wallet

Fast and private age verification

Given many sites and services require age verification, we wanted to develop a system that not only verifies age, but does it in a way that protects your privacy. That's why we are integrating Zero Knowledge Proof (ZKP) technology into Google Wallet, further ensuring there is no way to link the age back to your identity. This implementation allows us to provide speedy age verification across a wide range of mobile devices, apps and websites that use our Digital Credential API.

We will use ZKP where appropriate in other Google products and partner with apps like Bumble, which will use digital IDs from Google Wallet to verify user identity and ZKP to verify age. To help foster a safer, more secure environment for everyone, we will also open source our ZKP technology to other wallets and online services.

Language of Truth

The high-level goal: instead of relying on digital middlemen to answer queries or share data, Internet *users* can do this themselves!

“Language of Truth” Applications

- Move your likes, friends, followers, history, and reputation seamlessly across Facebook, Twitter, Reddit, Instagram, etc.
- Replace Equifax, Transunion, or Experian with cryptographic protocols; your smartphone can honestly answer any query about your financial history in a privacy-preserving way
- Universal + interoperable digital identity standards accepted by all websites, businesses, government agencies, universities, and more.

A Universal Protocol

- A single Protocol for all of your social data, digital identity, financial history, professional interactions, medical data, ...
- Every website on the Internet handles and outputs data in a way that is compatible with the Universal Protocol
- Every website on the Internet can verify, understand, and consume data conforming to the protocol

Interested in getting involved?

At 0xPARC, we work on:

- Research to advance the underlying cryptographic protocols.
- Prototyping and benchmarking new cryptographic schemes.
- Building open-source infrastructure for progcrypto applications.
- Developing and deploying the first applications of new crypto tech.

Reach out to brian@0xparc.org to learn more!

Thank You!