PureCoin

. . . and its lack of purity

Adam Gibson

12 Nov 2025

Bitcoin Historico

Why it's hard to stop data

- "Too technical" - no

Why it's hard to stop data

- "Too technical" - no
- Cypherpunk = replace trust with cryptographic verification

Why it's hard to stop data

- "Too technical" - no
- Cypherpunk = replace trust with cryptographic verification
- Cryptography $\neq$ encryption BUT obfuscated, random

Why it's hard to stop data

- "Too technical" - no
- Cypherpunk = replace trust with cryptographic verification
- Cryptography $\neq$ encryption BUT obfuscated, random
- High entropy: public keys signatures ...

Why it's hard to stop data

- "Too technical" - no
- Cypherpunk = replace trust with cryptographic verification
- Cryptography $\neq$ encryption BUT obfuscated, random
- High entropy: public keys signatures ...
- Embedding WORKS (there's space!)

Why it's hard to stop data

- "Too technical" - no
- Cypherpunk = replace trust with cryptographic verification
- Cryptography $\neq$ encryption BUT obfuscated, random
- High entropy: public keys signatures ...
- Embedding WORKS (there's space!)
- c.f. SWIFT

Why it's hard to stop data

- "Too technical" - no
- Cypherpunk = replace trust with cryptographic verification
- Cryptography $\neq$ encryption BUT obfuscated, random
- High entropy: public keys signatures ...
- Embedding WORKS (there's space!)
- c.f. SWIFT

Let's try to build PureCoin

PureCoin

- Only signed transfers, no scripts, no timelocks, no witness discount

PureCoin

- Only signed transfers, no scripts, no timelocks, no witness discount
- Pubkeys cannot be fake!

PureCoin

- Only signed transfers, no scripts, no timelocks, no witness discount
- Pubkeys cannot be fake!
- . . . means 3x longer addresses: P, R, s

PureCoin

- Only signed transfers, no scripts, no timelocks, no witness discount
- Pubkeys cannot be fake!
- . . . means 3x longer addresses: P, R, s

Let's see if data can still be embedded.

How to publish a secret

- $s = k + ex$

How to publish a secret

- $s = k + ex$
- $k = x =$ “data data data . . .”

How to publish a secret

- $s = k + ex$
- $k = x =$ “data data data . . .”
- extract data: $\frac{s}{1+e}$

How to publish a secret

- $s = k + ex$
- $k = x =$ "data data data ..."
- extract data: $\frac{s}{1+e}$

Leaks the key. Or ...?

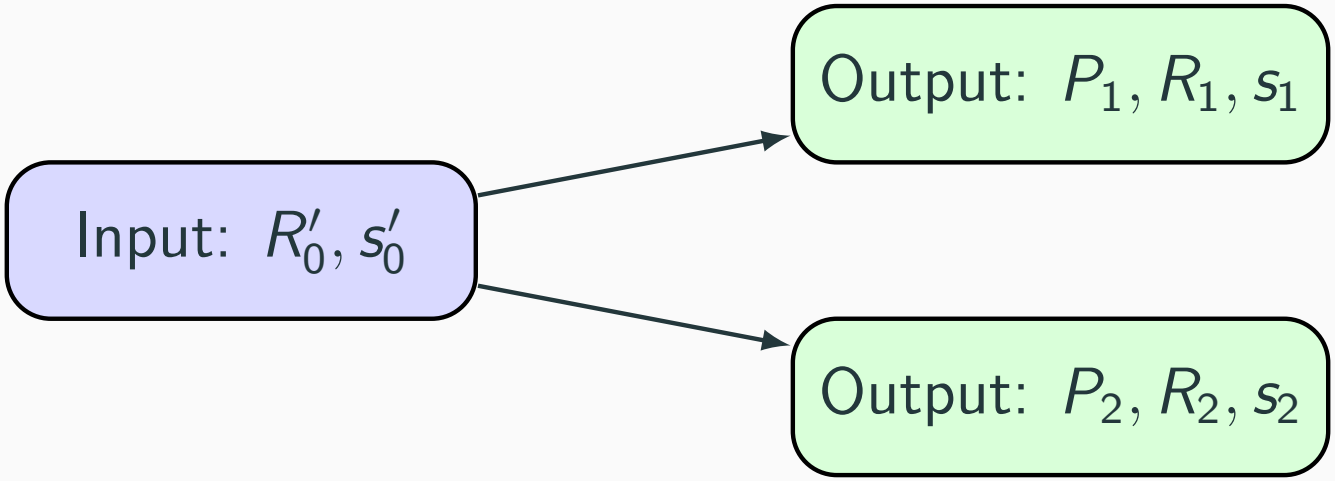
How to publish a secret

- $s = k + ex$
- $k = x =$ "data data data ..."
- extract data: $\frac{s}{1+e}$

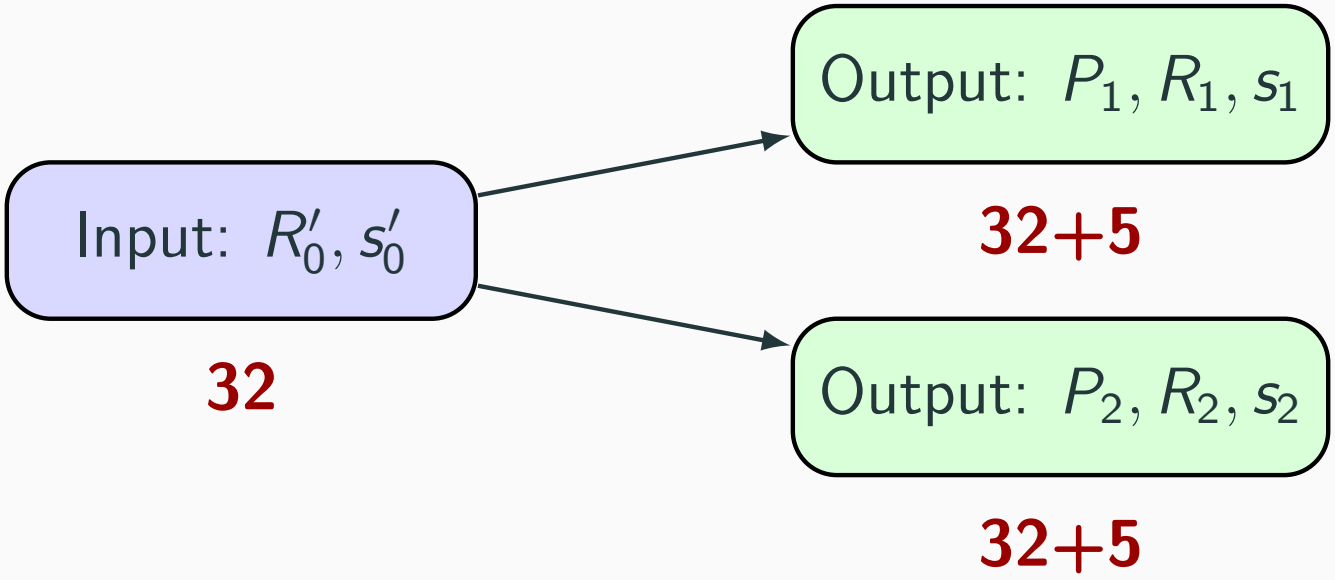
Leaks the key. Or ...?

See mailing list post on Schnorr unembeddability, paper.

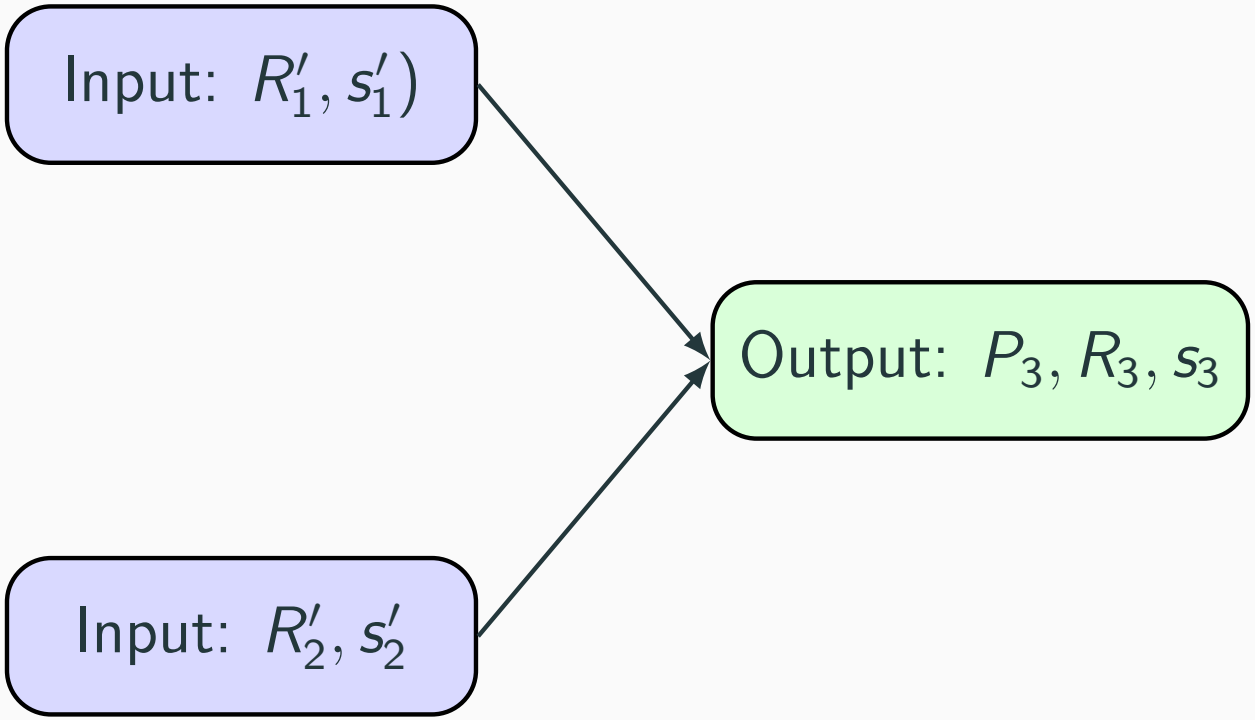
Transaction Flow - produce



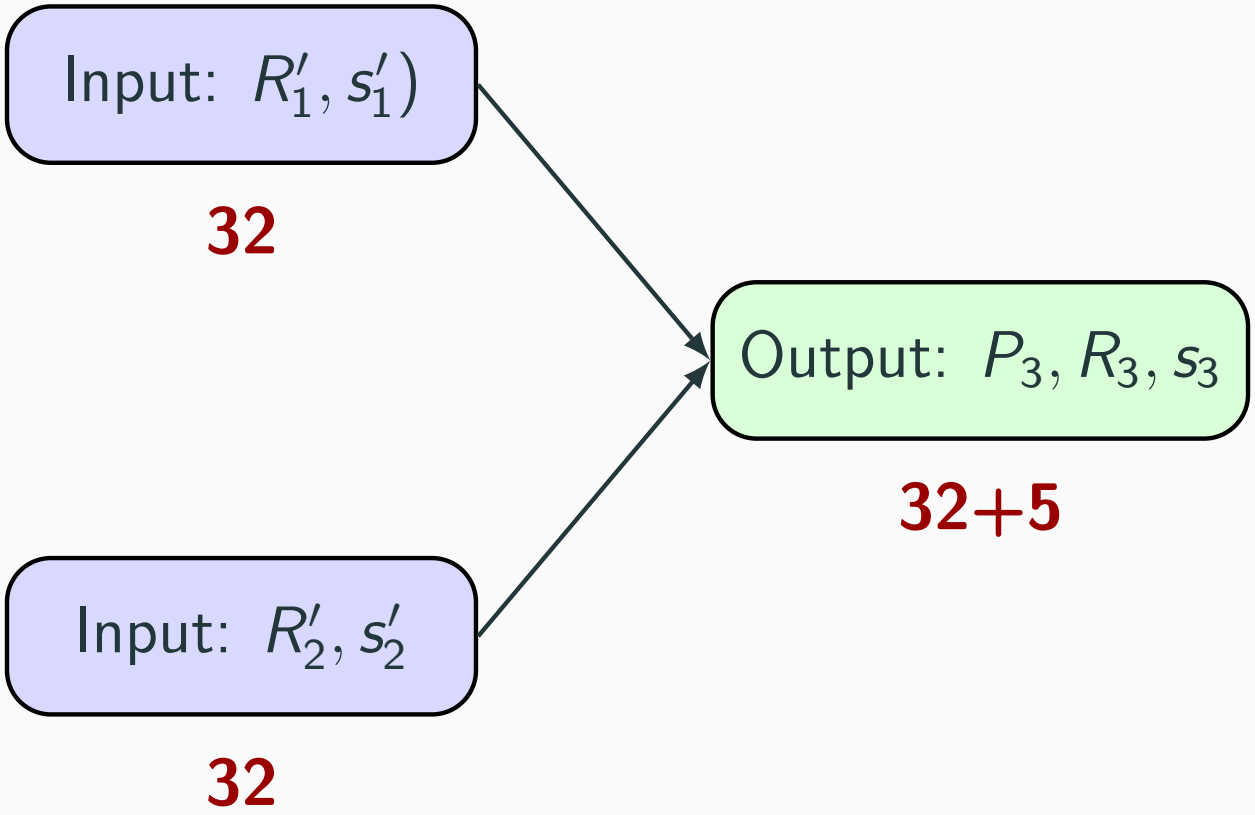
Transaction Flow - produce



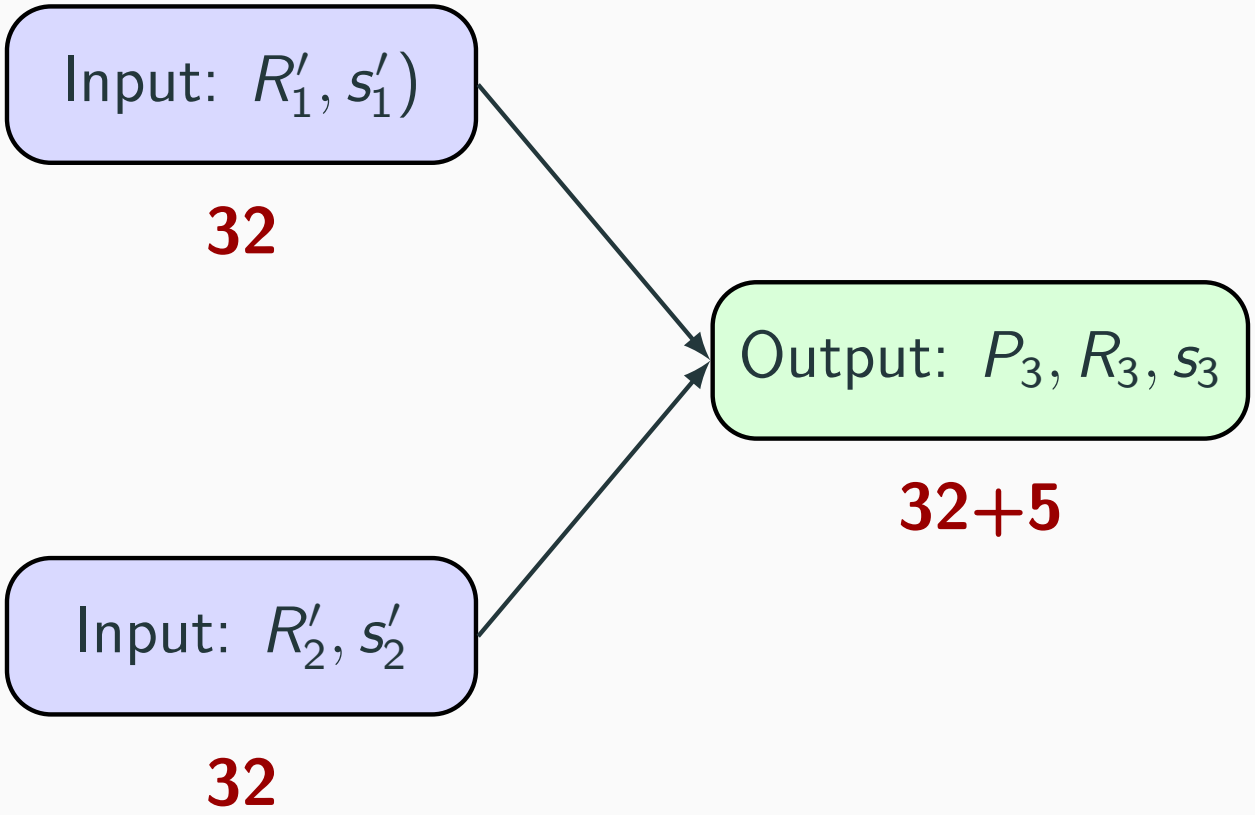
Transaction Flow - consume



Transaction Flow - consume



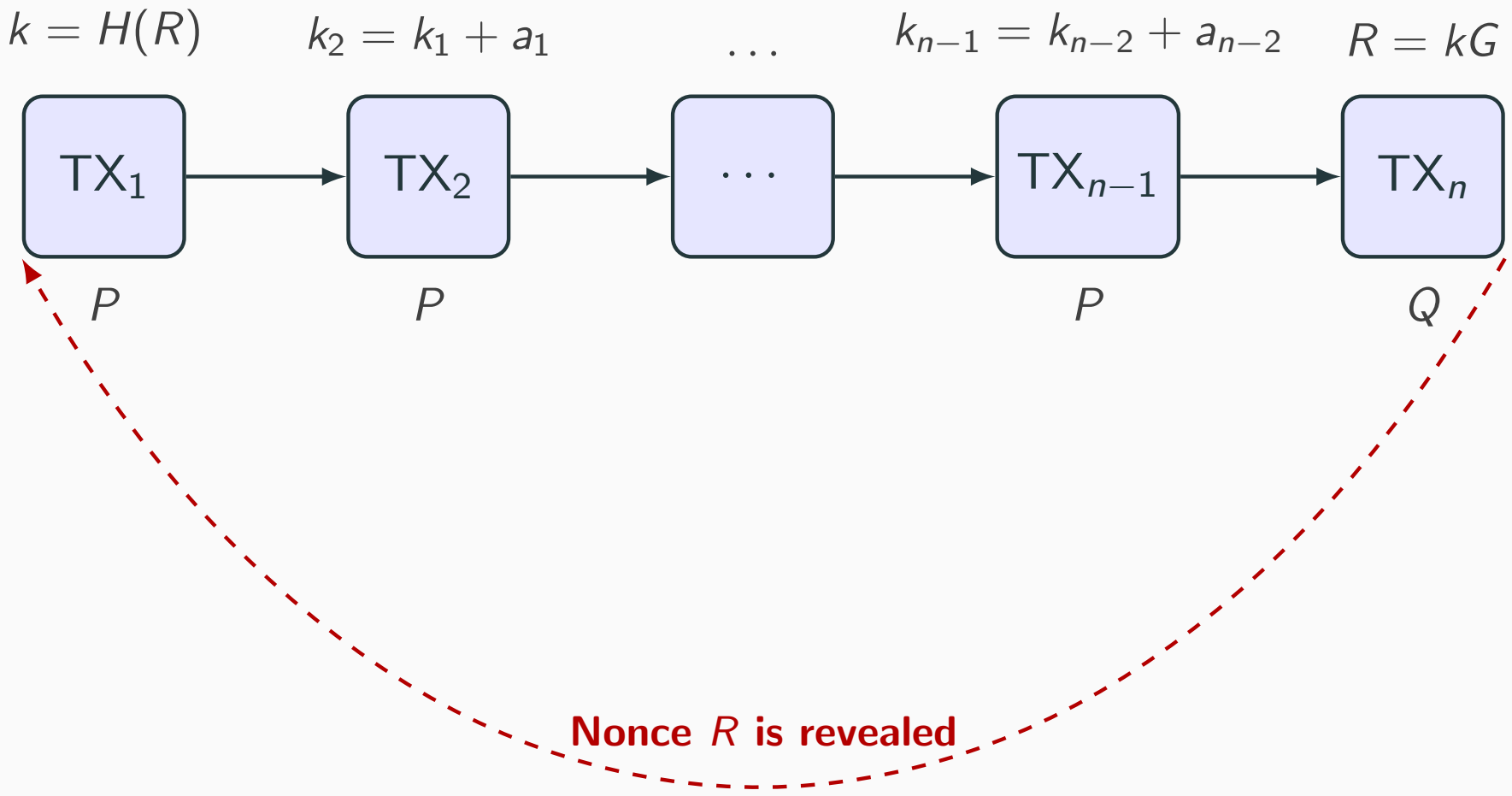
Transaction Flow - consume



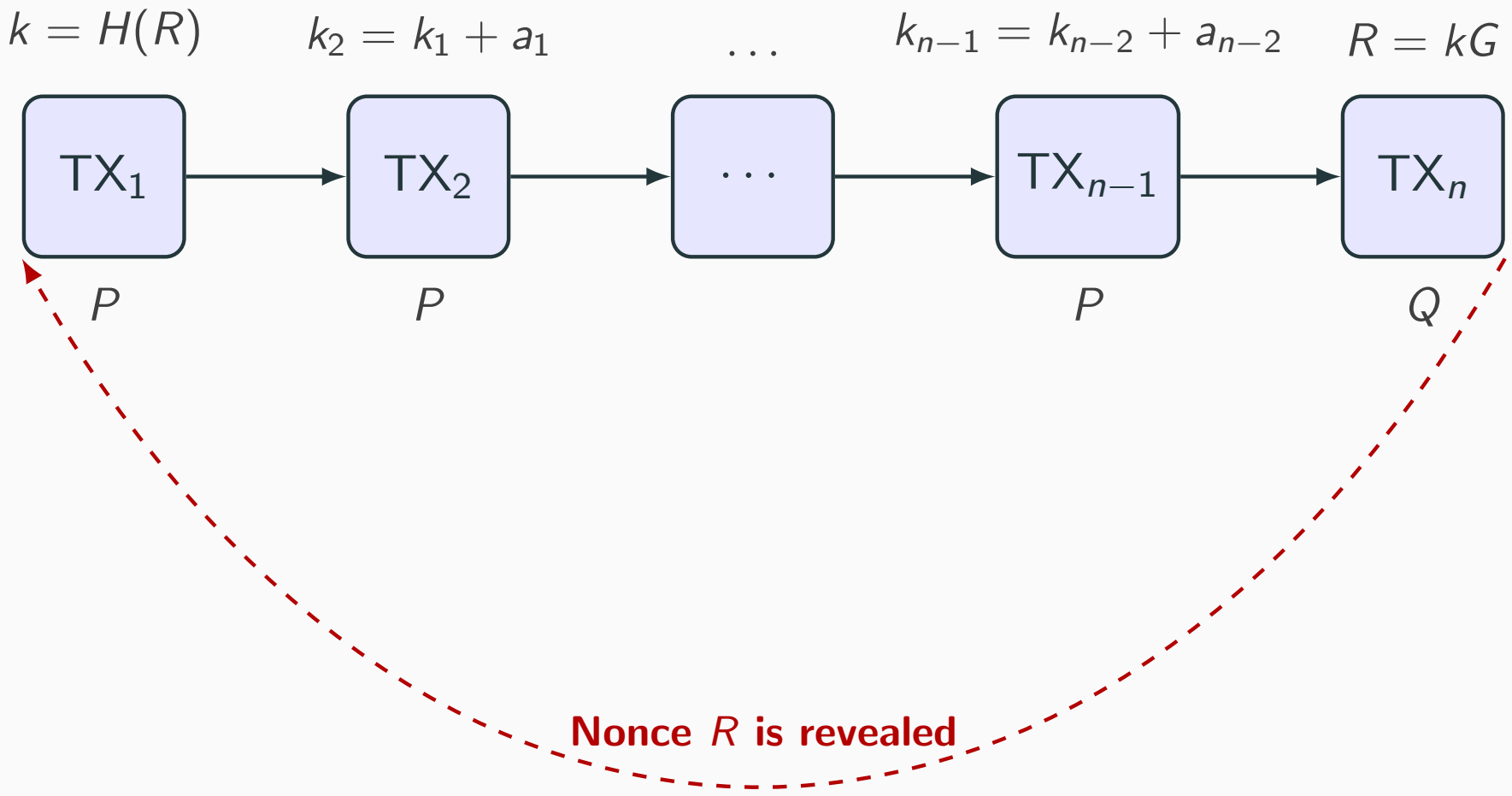
Total embedding ratio: $(3 \times 32 + 5)/(7 \times 32) =$

45%

Without leaking the private key, please!



Without leaking the private key, please!



h/t ajtowns

Assessment : is PureCoin viable?

- It's a (very) hard fork

Assessment : is PureCoin viable?

- It's a (very) hard fork
- It makes outputs about 3 times as expensive

Assessment : is PureCoin viable?

- It's a (very) hard fork
- It makes outputs about 3 times as expensive
- It removes time locks, swaps etc.

Assessment : is PureCoin viable?

- It's a (very) hard fork
- It makes outputs about 3 times as expensive
- It removes time locks, swaps etc.
- Lose all Layer 2? (Not certain!)

Assessment : is PureCoin viable?

- It's a (very) hard fork
- It makes outputs about 3 times as expensive
- It removes time locks, swaps etc.
- Lose all Layer 2? (Not certain!)
- **It doesn't remove data embedding at all**

Assessment : is PureCoin viable?

- It's a (very) hard fork
- It makes outputs about 3 times as expensive
- It removes time locks, swaps etc.
- Lose all Layer 2? (Not certain!)
- **It doesn't remove data embedding at all**

"ZK requires blinding which requires randomness. And data can fit into randomness".

– me, I just made it up

Secrecy $\Longrightarrow$ data ...?

- Another example: R is generated by RFC6979

Secrecy $\Longrightarrow$ data ...?

- Another example: R is generated by RFC6979
- To *prove* it, use a SNARK over HMAC-SHA256

Secrecy $\implies$ data ...?

- Another example: R is generated by RFC6979
- To *prove* it, use a SNARK over HMAC-SHA256
- Cool, so it can't be data-embedding right?

Secrecy $\implies$ data ...?

- Another example: R is generated by RFC6979
- To *prove* it, use a SNARK over HMAC-SHA256
- Cool, so it can't be data-embedding right?
- Right, but the SNARK can!

Secrecy $\implies$ data ...?

- Another example: R is generated by RFC6979
- To *prove* it, use a SNARK over HMAC-SHA256
- Cool, so it can't be data-embedding right?
- Right, but the SNARK can!
- (e.g. Groth16, r_C can embed data if r_A, r_B are public)

Secrecy $\implies$ data ...?

- Another example: R is generated by RFC6979
- To *prove* it, use a SNARK over HMAC-SHA256
- Cool, so it can't be data-embedding right?
- Right, but the SNARK can!
- (e.g. Groth16, r_C can embed data if r_A, r_B are public)
- BUT - BLS, deterministic proofs (not ZK)

Thank you

Contact info:

waxwing on nostr:
npub1vadcfln4ugt2h9ruwsuwu5vu5am4xaka7pw6m7axy79aqyhp6u5q9knuu7

https://github.com/AdamISZ

A blog: https://reyify.com/blog/

gpg: My 4668 .... 197A key is lost!

waxwing

npub1vad:5q9knuu7