Cryptanalysis of Witness Encryption: A Path to Bitcoin Privacy

Markus Schofnegger

TUM Blockchain Conference, Munich 2026

[[alloc] init]

Motivation

- Bitcoin payments are fully public
  - Amounts, addresses, transaction graph
- Goal is private payments natively on Bitcoin
  - No soft fork, no trusted operator
- Common on Ethereum, missing on Bitcoin
  - Private payments, bridges, rollups, ... enforced by smart contracts

The Ethereum Way



- The contract holds the funds and verifies the proof on-chain
- The proof does not reveal which deposit is withdrawn

Shielded Bitcoin – What We Want



- Private payments with real BTC, as on Ethereum[1]
  - Nobody sees amounts, sender or recipient
  - ZK proofs show that every payment follows the rules
- No soft fork, and nobody in the middle can run off with the BTC

[1] Shikhelman et al., Shielded Bitcoin: Private Transfers on the Bitcoin L1

Not Easy on Bitcoin

- Bitcoin Script is limited
  - No advanced arithmetic
  - SNARK verifier would fill many blocks, at huge cost
- No covenants
  - Opcodes like `OP_CAT` or `OP_CTV` would help, but need a soft fork
  - Debated for years, not activated

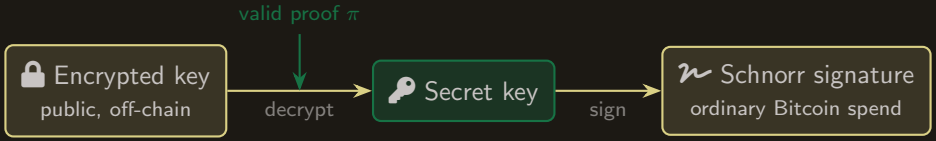
But Bitcoin can check signatures efficiently!

The Missing Link



- The BTC should be released only for a valid proof
- But Bitcoin cannot check the proof, only a signature
- Who should hold the signing key?
  - ✗ A trusted operator
  - ✓ Nobody, until someone presents a valid proof

Bitcoin PIPEs – A Key Locked Behind a Proof



- Setup: Encrypt a secret key with witness encryption[2]
- Lock: Send BTC to the matching public key
- Unlock: Valid proof decrypts the key

→ On-chain, just an ordinary signature

[2] Abdalla et al., Bitcoin PIPEs v2: Covenants and ZKPs on Bitcoin via Witness Encryption

Witness Encryption – The Crucial Component

Witness Encryption [GGSW13]

- $\mathsf{ct} \leftarrow \mathsf{Enc}(x, \mathtt{msg})$ encrypts a message `msg` under a statement x
- $\mathsf{Dec}(\mathsf{ct}, w)$ returns `msg` for any valid witness w for x
- Without a witness, ct reveals nothing about `msg`

- Encryption needs no witness and no interaction
- For PIPEs, `msg` is the secret signing key and x says "there is a valid proof π"
  - Verifying a SNARK is a fixed statement, however complex the rules are

→ We need efficient witness encryption for a SNARK verifier

How can we build efficient witness encryption?

Witness Encryption in the Literature

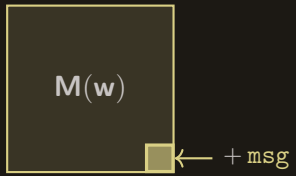
- We need witness encryption for a generic SNARK verifier
  - Large circuit with many gates
- Previous schemes work bit by bit, so a verifier needs $\approx 10^9$ gates
- Ours works with field elements, so ≈ 15000 constraints are enough

	[VWW22]	[BIJ+20]	Ours (first version)	Ours (current)
Ciphertext size	10^{77} TB	10^{14} TB	338 TB	≈ 1 TB

Core Idea – A Message Behind a Rank Drop

- Turn the statement into a matrix $\mathbf{M}(\mathbf{x}) = \sum_i x_i \mathbf{M}_i$
  - An affine determinant program (ADP) [BIJ+20]
  - Valid witness → $\mathbf{M}(\mathbf{x})$ is singular
  - Anything else → $\mathbf{M}(\mathbf{x})$ is invertible
- Encrypt – add `msg` to a corner, publish all $\mathbf{M}_i$
- Decrypt – plug in a valid witness
  - The rank drop reveals `msg`

→ Only linear algebra



From a Constraint to a Gate

- Write the statement as m quadratic constraints in the input $\mathbf{x} = (1, x_1, \ldots, x_n)$

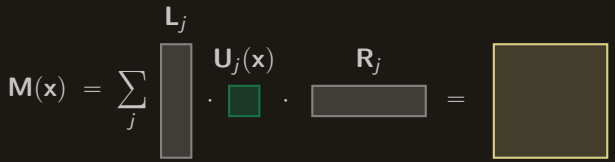
$$a_j \cdot b_j = c_j \cdot d_j, \quad \text{where } a_j, b_j, c_j, d_j \text{ are linear in } \mathbf{x}$$

- Each constraint becomes a 4×4 gate

$$\mathbf{U}_j(\mathbf{x}) = \begin{pmatrix} a_j & c_j & -\xi_j & 0 \\ d_j & b_j & 0 & \xi_j \\ 0 & 0 & b_j & c_j \\ 0 & 0 & d_j & a_j \end{pmatrix}$$

- $\operatorname{rank}(\mathbf{U}_j) = 2$ if the constraint holds and 4 if it does not
- a_j, b_j, c_j, d_j are from the constraint, ξ_j is a random linear form

From Gates to a Ciphertext



- One gate per constraint, each mixed with secret random $\mathbf{L}_j$ and $\mathbf{R}_j$
- Valid witness → every gate has rank 2 → the sum is singular
- Wrong input → some gate has rank 4 → the sum is invertible

What can go wrong?

- The attacker gets all matrices $\mathbf{M}_0, \ldots, \mathbf{M}_n$
- Real circuits are sparse, each variable appears in only a few constraints
  - Then every $\mathbf{M}_i$ has a special shape

$$\mathbf{M}_i = \mathbf{L} \cdot \Big(\underbrace{\text{diagonal}}_{\text{from the masks}} + \underbrace{\text{low rank}}_{\text{from a few gates}} \Big) \cdot \mathbf{R}$$

- $\mathbf{L}$ and $\mathbf{R}$ are secret, but the same for every i

$\rightarrow$ Random matrices do not have this shape

⇄ Detecting the Shape

- A short linear-algebra test based on commutators measures this shape

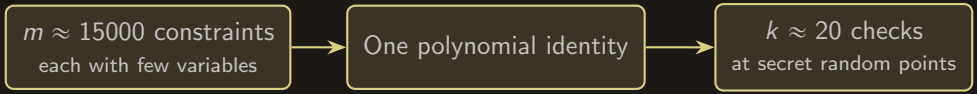
Input to the test	**Rank** (at most 33)
Ciphertext of a sparse circuit	14 to 18
Random matrices	33

→ The ciphertext is distinguishable from random

- Message can also be recovered without a witness
- Breaks the first version of our scheme[3]

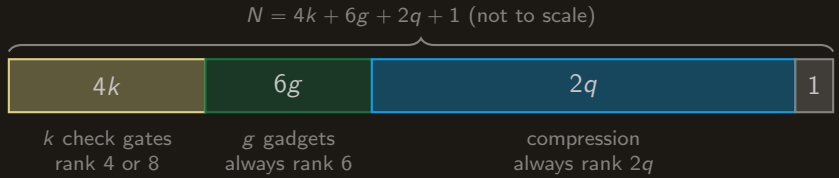
[3] Gebali et al., "Breaking ADP-Based Witness Encryption"

Countering Sparsity – Dense Checks at Secret Points



- Every variable enters every check, so nothing is sparse anymore
  - The commutator attack no longer works
- Each check becomes one gate
  - Same as before but with 8×8 gate

Current Design



- Only the k check gates change their rank for a wrong witness
- With so few checks, the public matrices carry too little randomness
  - Other linear-algebra attacks could exploit this
  - Gadgets and compression add randomness that does not depend on the circuit
- Ciphertext of $\approx$ 1 TB (preliminary)
  - $k \approx 16$ to 24, $g = 10$, $q \approx 400$, for ≈ 30000 variables

Cryptanalysis Directions

Attack idea	Status
Matrix identities, e.g. commutators	Stopped by the dense checks
Too little randomness in the matrices	Stopped by gadgets and compression
Inputs where the rank drops (MinRank)	Too expensive, sets the parameters
Solving for the secrets directly	Far too expensive

→ Every part of the current design answers a concrete attack

❓ Still Open

- So far, no attack on the current design
- No reduction from a standard assumption
  - Security relies on cryptanalysis
- Cost estimates for the new components still need validation
- Smaller ciphertexts via smaller verifier circuits or batching

Many Use Cases

- Any spending condition you can prove becomes a Bitcoin spending condition
  - Where an Ethereum contract verifies a proof, a PIPE can lock BTC instead

On Ethereum	On Bitcoin with PIPEs
Private payments	Shielded Bitcoin
ZK rollups and bridges	BTC released on a proof about another chain
Conditional vaults	BTC that unlocks only if a rule is proven

- No soft fork, the proof is checked off-chain inside the decryption

Third-Party Analysis

- New paper with the full construction and our cryptanalysis coming soon
- Confidence in heuristic schemes only comes from public cryptanalysis
  - The first version was broken by exactly this kind of analysis
- Please try to attack it!
  - Distinguishers, structure that "should not be there"
  - Better attacks or cost estimates for MinRank and flattenings

🏁 Summary



- Private payments with real BTC, as on Ethereum
- PIPEs turn a valid proof into an ordinary signature, no soft fork needed
- The crucial component is efficient witness encryption
- Enables many other use cases on Bitcoin

Resources

- Shielded Bitcoin
  - allocinit.xyz/uploads/shielded-bitcoin.pdf
- Bitcoin PIPEs v2
  - ia.cr/2026/186
- Details about breaking first construction
  - ia.cr/2026/1594
- New witness encryption paper – coming soon!



ia.cr/2026/1594

allocinit.xyz @allocinitxyz

Part I

Appendix

The 8×8 Check Gate

$$\mathbf{U} = \left(\begin{array}{cccc|cccc} -b & d & f & 0 & \xi & 0 & 0 & 0 \\ -c & a & 0 & f & 0 & \xi & 0 & 0 \\ -e & 0 & a & -d & 0 & 0 & \xi & 0 \\ 0 & -e & c & -b & 0 & 0 & 0 & \xi \\ \hline 0 & 0 & 0 & 0 & a & -d & -f & 0 \\ 0 & 0 & 0 & 0 & c & -b & 0 & -f \\ 0 & 0 & 0 & 0 & e & 0 & -b & d \\ 0 & 0 & 0 & 0 & 0 & e & -c & a \end{array}\right) = \begin{pmatrix} \mathrm{A} & \xi \mathbf{I}_4 \\ \mathbf{0} & \mathrm{B} \end{pmatrix}$$

- $\mathrm{AB} = \mathrm{BA} = -(ab - cd - ef)\,\mathbf{I}_4$ and $\operatorname{rank}(\mathbf{U}) = 4 + \operatorname{rank}(\mathrm{BA})$
  - Rank 4 if the check holds, rank 8 if it does not, and never below 4
- $e = Z(\tau)\,S(\sqrt{\tau})$ and $f = S(-\sqrt{\tau})$, so $S(\sqrt{X}) \cdot S(-\sqrt{X}) = H(X)$ gives $e \cdot f = Z(\tau) \cdot H(\tau)$
  - Not every H has such an S, rerandomize a few witness bits until one does

References

[ACG+26] Michel Abdalla, Brent Carmer, Muhammed El Gebali, Handan Kilinc-Alper, Mikhail Komarov, Yaroslav Rebenko, Lev Soukhanov, Erkan Tairi, Elena Tatuzova, and Patrick Towa. *Bitcoin PIPEs v2: Covenants and ZKPs on Bitcoin via Witness Encryption*. IACR Cryptol. ePrint Arch., Paper 2026/186. 2026.

[BIJ+20] James Bartusek, Yuval Ishai, Aayush Jain, Fermi Ma, Amit Sahai, and Mark Zhandry. *Affine Determinant Programs: A Framework for Obfuscation and Witness Encryption*. In: ITCS. Vol. 151. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020, 82:1–82:39.

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. *Witness Encryption and its Applications*. In: STOC. ACM, 2013, pp. 467–476.

[GRSS26] Muhammad El Gebali, Yaroslav Rebenko, Markus Schofnegger, and Lev Soukhanov. *Breaking ADP-Based Witness Encryption*. In: IACR Cryptol. ePrint Arch. 2026 (2026), p. 1594.

[SKM26] Clara Shikhelman, Mikhail Komarov, and Aleksei Moskvin. *Shielded Bitcoin: Private Transfers on the Bitcoin L1*. 2026.

[VWW22] Vinod Vaikuntanathan, Hoeteck Wee, and Daniel Wichs. *Witness Encryption and Null-IO from Evasive LWE*. In: ASIACRYPT (1). Vol. 13791. Lecture Notes in Computer Science. Springer, 2022.