Redeth Pay: Private Stablecoin Payroll with a Complete Audit Trail

September 2026 · Draft v0.1

Abstract

Organizations that pay salaries and suppliers in USDC publish every payee, amount and date to anyone with a block explorer. Redeth Pay is a settlement rail on Base that encrypts the contents of each payment on the public chain while keeping every payment legible to the paying organization and its accountant. Payments are notes committed to a Merkle tree and spent with zero-knowledge proofs generated on the payer's device and verified on chain. A coordinator batches those proofs but never holds funds or keys, cannot alter or decrypt a payment, and can be bypassed: any user can exit the pool without it. Note contents are encrypted to a recipient viewing key. A separate viewing key held by the organization's treasury account decrypts what it received and, through sender receipts, its outflow ledger, without asking any employee for their keys. This paper explains why the product exists and how it works.

Terms

Term	Description
Note	A private record of value: asset, amount, owner and randomness. A payment creates notes; spending consumes them.
Commitment	The hash of a note, published on chain in place of the note itself.
Merkle tree, state root	A hash tree holding every commitment. The contract stores only its root, and a proof can show that a note is in the tree without saying which one.
Nullifier	A value published when a note is spent, derived so that it marks the note as used without revealing which note it was.
Zero-knowledge proof	A proof that a statement is true, such as "I own a note in the tree and the amounts balance", that reveals nothing beyond the statement. The pool contract checks the proof instead of seeing the payment.
Batch	A group of payments proven together and settled in one transaction; sixteen in the current design.
Encrypted payload	The contents of a note, encrypted to its recipient and published on chain beside the commitment.
Change note	The note a payer sends back to itself with whatever is left after a payment. Every payment creates one, even when nothing is left.
Sender receipt	A record of an outgoing payment (payee, amount, memo, date) that the payer's wallet stores inside its own change note.
Viewing key	A key that decrypts the notes addressed to an account. It cannot spend.

Term	Description
Nullifying key	A key that derives an account's nullifiers, so a holder can detect when that account's notes are spent. It cannot spend.
Viewing-key share	A read-only credential built from an account's viewing key, optionally with its nullifying key, that the account owner hands to an auditor.
Registry	The on-chain directory that maps a username to the keys a payer needs to pay it.
Coordinator	The off-chain service that collects proofs into batches and submits them to the pool contract.
Escape hatch	A withdrawal path that works without the coordinator, for when it stops or refuses service.

1. Why: payroll moved on chain and stayed public

McKinsey and Artemis estimate that stablecoins settled about $390 billion of real payments in 2025, more than double the 2024 figure, of which about $226 billion was business-to-business, up 733 percent in a year, and about $90 billion payroll and remittances [1]. Payroll platforms have followed. Rise has processed more than $1 billion in payroll with over half of worker withdrawals in stablecoins [2]. Toku reports more than $1 billion a year in stablecoin payroll [13], and Deel lets salaried employees take part of their net pay in USDC [3]. Stablecoin supply was about $315 billion in March 2026 [2], and Citi's base case puts it at $1.9 trillion by 2030 [4].

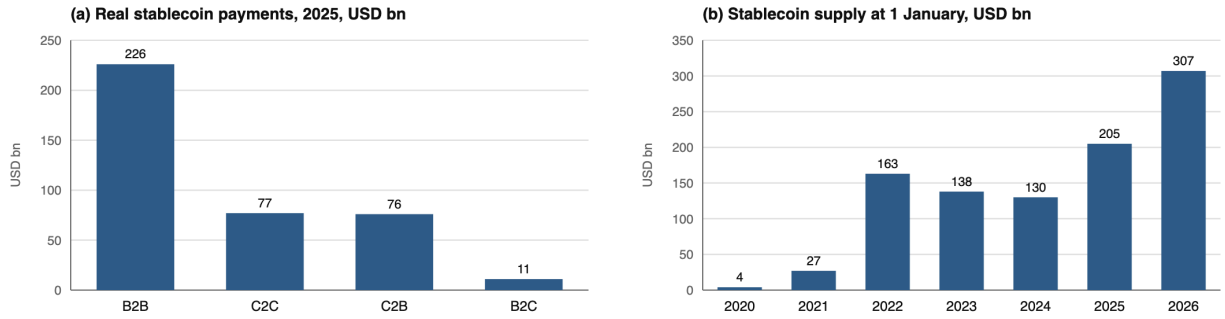

Figure 1. (a) Stablecoin payments in 2025 by segment, after removing trading and internal transfers; the four segments sum to $390 billion [1]. (b) Total stablecoin supply at the start of each year [9]; $312 billion on 12 September 2026. Citi's base case for 2030 is $1.9 trillion, bull case $4 trillion [4].

Most of that payroll settles as a plain token transfer. The transaction shows the payee's address, the amount and the timing, and a monthly pattern of identical amounts reads as a salary to anyone looking. This leaks pay equity inside a team and contractor rates and vendor pricing to competitors, and compensation linked to an identifiable employee is personal data: under GDPR the employer must protect it with security appropriate to the risk [10], and under California's privacy law, which has covered employee data in full since 2023, covered businesses must apply reasonable security to it [11]. The European Data Protection Board's 2026 guidelines on blockchain say that storing personal data on a blockchain should be avoided where it conflicts with data protection principles, and name encryption and

cryptographic commitments as the techniques to use when on-chain storage is necessary [12]. The teams most exposed are the ones already on chain: DAOs, grants programs and remote-first companies paying contractors in USDC, whose payroll today is often a Safe multisig and a spreadsheet.

Private payment rails exist, and private payroll has shipped on some of them. Toku, with Aleo and Paxos Labs, has run private stablecoin payroll in production since January 2026: settlement happens on the Aleo chain in USAD, a Paxos Labs dollar, and the employer keeps an exportable payroll record for tax and audit [16]. Aleo also carries USDCx, a dollar backed one-to-one by USDC through Circle's xReserve [16]. Zama's confidential USDC and USDT on Ethereum encrypt amounts and balances with fully homomorphic encryption while sender and recipient addresses stay public, decryption is handled by a threshold key-management network, and one company has paid salaries through it [17, 19]. Hinkal runs a zero-knowledge privacy layer with stealth addresses on Ethereum, Base and several other chains, with per-wallet viewing keys and Chainalysis screening, and offers it to payroll platforms as an SDK [18]. Railgun offers shielded transfers on Ethereum, BSC, Polygon and Arbitrum with view-only wallets [5], and Aztec is a privacy-first Layer 2 on Ethereum [6].

Hinkal is the closest comparison: a zero-knowledge privacy layer on Base that keeps the payer's wallet and native USDC. Redeth Pay is built for one job inside that space, an organization's payroll, and that shows in three places. The ledger: a sender receipt binds each payment's payee and amount to the on-chain commitment, and an auditor holding the treasury's nullifying key can detect every spend from public chain data, so the finance lead gets a record that can be checked rather than a wallet's history. The exit: unspent funds can leave the pool without the coordinator through a 24-hour exit that neither the operator nor the admin can block. The pay run: payees are addressed by username, a run settles in parallel batches with each line marked paid only at chain finality, and approvals stay in the application while a multisig funds the treasury. Against Toku on Aleo and Zama the differences are structural: no new chain, no wrapped token, and no party on the network holding a key that can decrypt a payment [16, 17]. What Redeth Pay does not do is payroll itself: tax, payslips and off-ramps stay with the payroll platform.

	Hidden on chain	**Audit view**	**Who can decrypt a payment**	**Payee addressed by**	**Settles on**
Plain USDC transfer	Nothing	Public ledger, visible to everyone	Everyone	Address	Every chain where USDC is issued, native USDC
Toku on Aleo	Payee and amount	Exportable payroll record kept by the employer, produced through the provider's platform; selective disclosure on Aleo	The employer, and the parties it discloses to	Address	Aleo, in USAD; USDCx also available on Aleo

	Hidden on chain	Audit view	Who can decrypt a payment	Payee addressed by	Settles on
Zama confidential tokens	Amounts and balances; sender and recipient addresses stay public	Access-control list grants decryption to named parties	The holder, and parties admitted through the protocol's threshold key network	Address	Ethereum, in cUSDC or cUSDT, confidential versions of USDC and USDT
Hinkal	Payee and amount	One wallet, via its viewing key, full or partial history	The wallet owner and anyone holding its viewing key	Address	Ethereum, Base, Polygon, Arbitrum, Optimism, Solana, Tron and others
Railgun	Payee and amount	One wallet, via its view-only key	The wallet owner and anyone holding its view-only key	Address	Ethereum, BSC, Polygon, Arbitrum
Redeth Pay	Payee, amount and memo	The organization's treasury account, via a viewing-key share that cannot spend; outflows carry payee, memo and date through sender receipts, and a share that includes the nullifying key lets the auditor check every spend against chain data	The recipient's viewing key, and the payer's own for its change note and receipt; no network party	Username	Base, native USDC; Ethereum L1 supported

Table 1. What a paying organization gets from each option, from each project's own documentation [5, 16, 17, 18, 19]. Aztec is a general privacy Layer 2 and is left out of the table; its keys page lists outgoing viewing keys as reserved and not yet used [6].

2. Design overview

Redeth Pay is a shielded pool contract on Base holding USDC, a coordinator that batches proofs, and clients that prove on the user's device. The construction follows the note-and-nullifier model introduced by Zcash [7], with the engineering choices below.

Notes and commitments. A payment is a note `(asset_id, amount, owner_pubkey, rho, r)` over the BN254 scalar field. Its Poseidon commitment is inserted into a depth-32 sparse Merkle tree in the pool contract. Spending a note publishes a nullifier that the contract records so the note cannot be spent twice.

The nullifier is derived from a nullifying key, the note commitment and the note's canonical tree position, so uniqueness rests on the hash instead of on the sender choosing fresh randomness.

Keys. A user's spending key and viewing key are derived from a single wallet signature, and the nullifying key from the spending key and the chain id. There is no separate secret to back up: the wallet that produced the signature can reproduce the keys, which requires a signer that always yields the same signature for the same message. The spending key spends. The viewing key decrypts every payload addressed to it and cannot spend, forge a nullifier or reveal the spending key.

Transfers. A transfer is a 1-in, 2-out action: it consumes one note and emits a recipient note and a change note. It always emits both, so exact and partial payments look identical on chain. The sender proves in the browser, over Halo2 with KZG on BN254, that the input exists in the tree, that they own it, and that value is conserved. Sixteen actions form a batch. The coordinator proves the batch, commits the new state root and the encrypted payloads, and the contract verifies every proof on chain.

Encryption. Each output note is encrypted to the recipient's viewing public key: an ephemeral BabyJubJub ECDH agreement yields a symmetric key, and the note is sealed with AES-256-GCM under a nonce bound to the note commitment. A 16-bit view tag lets a wallet skip payloads that are not addressed to it without decrypting. Every payload is padded to one fixed 722-byte class, enforced by the pool contract, so its length reveals nothing about the presence of a memo or receipt.

Usernames. A registry contract maps a human-readable username to the recipient's keys. The payer types a name and never sees an address. The client reads the key tuple atomically to prevent key-substitution attacks.

Why Base. Base has carried native Circle USDC since 2023 [14] and has a large population of teams already paying contractors in USDC. The measured hashing component of settlement is under one cent per payment on Base, before proof verification and calldata, which are still to be measured. That cost is what makes a flat subscription with no per-payment fee possible. Ethereum L1 remains a supported deployment.

Redeth Pay launches with USDC only. The pool binds each asset id to one ERC-20 through an allowlist, so a second dollar stablecoin can be added without a protocol change. The likely candidate is Open USD, announced on 30 June 2026 by a consortium of more than 140 companies including Stripe, Visa, Mastercard and Coinbase, with fee-free minting and redemption and reserve earnings paid to participants less a management fee [8]. It is not yet issued or liquid, and Redeth Pay adds it once it is.

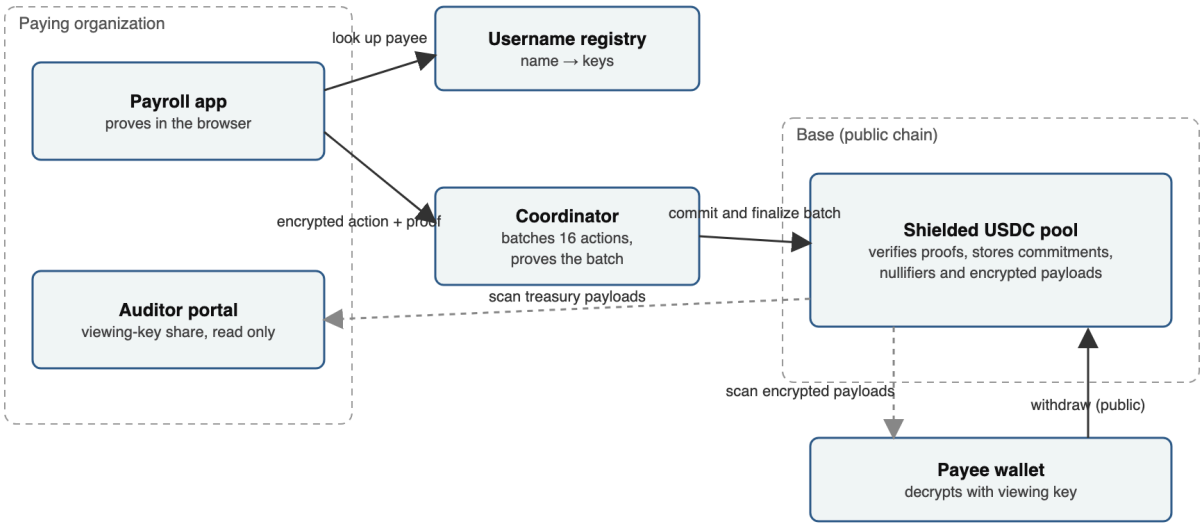

Figure 2. Data flow. Solid arrows carry proofs and encrypted payloads; dotted arrows are read-only scans with a viewing key. The coordinator never holds funds or keys.

3. The organization's view

What separates Redeth Pay from a personal privacy wallet is the organization's own view of its money.

The organization pays from a treasury account it owns. A multisig can fund that account. Approving a pay run is an application control, and the treasury account's own key signs each private payment. That is a weaker control than a multisig signature on every payment, and it is the price of spending privately. That account's viewing key decrypts its deposits and its change notes, and a sender receipt carried in each change-note ciphertext records each outgoing payment the wallet makes: payee, amount, memo, date and transaction reference. No employee's viewing or nullifying key is ever requested.

The finance lead or external accountant pastes a viewing-key share into a read-only auditor portal and exports CSV. The share is scoped to the organization's account and cannot spend.

Each payee sees "Received 4,000 USDC, August salary" and withdraws to a wallet, an exchange or an off-ramp provider. On-ramp, off-ramp, payslips and tax filing are left to payroll platforms such as Rise and Deel.

The payer's interface shows a payee list, amounts, a memo per line and one button, with notes, nullifiers, proofs and gas appearing nowhere outside the auditor portal. A pay run splits the treasury's balance into one note per payee, then settles the payments in parallel batches. A line is marked paid once its batch is final on chain.

The ledger does not rest on trusting the payer's software, provided the auditor holds the treasury account's genuine nullifying key. An auditor share can carry that key, which derives the nullifier of every note the treasury owns without being able to spend it. The auditor computes those nullifiers and matches them against the ones the pool contract has published, so every spend from the treasury is detected from public

chain data, and the amount of each spend follows from the value conservation the proof enforces. A receipt, when present, is checked against the on-chain output commitment, so it cannot name a false payee or amount. What the payer's wallet controls is only whether the payee name and memo are attached: a spend with no receipt shows in the audit as an outflow of known amount with no payee, an anomaly the auditor sees rather than a payment that can be hidden. Splitting the treasury's balance before a pay run also consumes notes, but the new notes still belong to the treasury, so the audit shows them as internal moves, not payments.

4. Privacy model and limitations

Hidden on chain: the payee, the amount and the memo of every private transfer, and whether a payment was exact or partial.

Public on chain: deposits into and withdrawals out of the pool, with amounts and addresses; registry usernames and keys; commitments, batch timing and occupancy.

A spend does not publish the consumed note's commitment. The action's public input is a sender-committed digest of the published ciphertext, so the published action data carries no direct identifier of the note being spent.

Metadata that remains: stable payroll timing, salary-sized public withdrawals, and which recent state root a sender proved against. Batch size is not an anonymity-set guarantee. The coordinator sees submission metadata and recipient keys. It is trusted for liveness, including publishing every batch's payloads, and for nothing else: it cannot take custody, alter value or decrypt.

5. Security, recovery and audit

Every proof is verified on chain, and the coordinator cannot authorize a spend or move funds. Every spendable note is recoverable from the wallet signature plus public chain data from an archive node, with payload bytes kept in calldata or logs so they never depend on blob retention. The exact ciphertext is bound into the sender's proof, so a coordinator that substitutes or omits a payload cannot finalize the batch, and cannot make a note unrecoverable.

Component	Trusted for	Cannot
Pool contract	Holding USDC, verifying every proof, enforcing the payload binding, recording commitments, nullifiers and encrypted payloads	Release funds without a valid proof, except refunding an unbatched deposit to its depositor
Circuits	Enforcing note ownership, tree membership, value conservation and nullifier derivation	Be satisfied by a false statement, assuming the proving setup below
Proving setup	The public Perpetual Powers of Tau ceremony on BN254; sound if at least one contributor was honest. Independent anchoring of the ceremony file's origin is a launch gate	—

Component	Trusted for	Cannot
Coordinator	Liveness: batching proofs and submitting batches	Take custody, alter or decrypt a payment, substitute a payload, or block a user's exit
Username registry	Mapping names to keys, append-only and not upgradeable	Change a name's keys without the owner's signature, outside a registry migration
Admin (at least 2-of-3 multisig, at least 24-hour timelock)	Configuration; pausing deposits and batch settlement; pausing withdrawal claims for at most seven days in any fourteen	Move funds, alter proofs, or pause escape, exit and refund paths
Chain and USDC issuer	Base or Ethereum staying live with its data available; Circle honoring USDC	Be bypassed: no exit path runs through a chain outage, and Circle keeps the power to block addresses and freeze USDC [15]

Table 2. Trust assumptions.

If the coordinator stops finalizing batches for seven days, anyone can activate an escape hatch and users withdraw against the last finalized root with a zero-knowledge proof that keeps the spending key secret. A global timeout does not protect a single censored user, so any unspent note can also be exited permissionlessly after a fixed 24-hour delay that no batch can reset, and an unbatched deposit can be refunded. The admin cannot pause any of these paths.

Production admin is at least a 2-of-3 multisig behind a timelock of at least 24 hours. The protocol specification maintains a threat inventory as a binding table: each identified attack class carries its mitigation or an accepted limitation, and every code-side mitigation has a test. Cryptographic constants and domain-separation tags are pinned by known-answer tests that every component must reproduce; verifier keys and contract addresses are pinned by a deployment manifest.

6. Selective disclosure and compliance

Privacy in Redeth Pay is privacy from the public, not from the parties with a right to know. Every deposit and withdrawal is public, with its address and amount, so banks, exchanges and regulators can screen funds entering or leaving the pool as they screen any USDC transfer. Inside the pool, disclosure is by key. The organization decrypts its own ledger with its treasury viewing key, and can hand a read-only share of it to an accountant, an auditor or a regulator, scoped to its own account and unable to spend. A payee can do the same for their own account. A share built from the viewing key alone can be retired for future payments by rotating that key. A share that also carries the nullifying key reveals the account's spends, past and future, and cannot be revoked, so it is issued only with separate consent. Nothing is disclosed by default, and nothing inside the pool is decrypted without its key.

References

1. Higginson, M., Zorrilla, A., Madden, J., Kirchner, M., McKinsey & Company with Artemis Analytics, "Stablecoins in payments: What the raw transaction numbers miss", 18 Feb 2026. https://www.mckinsey.com/industries/financial-services/our-insights/stablecoins-in-payments-what-the-raw-transaction-numbers-miss; segment figures from McKinsey Chart of the Week, "Stablecoins find their niche", 26 May 2026, https://www.mckinsey.com/featured-insights/charts/stablecoins-find-their-niche (accessed 2026-09-12)
2. Rise, "State of Crypto Payroll Report 2026", 23 Mar 2026. https://www.riseworks.io/blog/state-of-crypto-payroll-report-2026 (accessed 2026-09-10)
3. Deel Help Center, "Getting Paid in Stablecoins" (10 to 25 percent of net salary in USDC, EURC or USDT for EOR and direct employees). https://help.letsdeel.com/hc/en-gb/articles/42834641567121-Getting-Paid-in-Stablecoins (accessed 2026-09-12)
4. Citi Institute, "Stablecoins 2030: Web3 to Wall Street", 25 Sep 2025. https://www.citigroup.com/global/insights/stablecoins-2030 (accessed 2026-09-10)
5. Railgun documentation, "RAILGUN 101" and "View-Only Wallets". https://docs.railgun.org/wiki; https://docs.railgun.org/developer-guide/wallet/private-wallets/view-only-wallets (accessed 2026-09-12)
6. Aztec Network, "Aztec Documentation", overview and "Keys" page. https://docs.aztec.network/developers/overview; https://docs.aztec.network/developers/docs/foundational-topics/accounts/keys (accessed 2026-09-12)
7. Hopwood, Bowe, Hornby, Wilcox, "Zcash Protocol Specification". https://zips.z.cash/protocol/protocol.pdf (accessed 2026-09-10)
8. Open Standard, "Introducing Open USD", 30 Jun 2026. https://joinopenstandard.com/blog/introducing-open-usd (accessed 2026-09-12); member list as reported by The Block, "Visa, Stripe, Coinbase and more join Open USD stablecoin that shares reserve revenue", 30 Jun 2026, https://www.theblock.co/post/406736/visa-stripe-coinbase-join-open-usd-stablecoin-shares-reserve-revenue (accessed 2026-09-10)
9. DefiLlama, "Stablecoins" dashboard, total circulating supply, all chains. https://defillama.com/stablecoins; data from https://stablecoins.llama.fi/stablecoincharts/all (accessed 2026-09-12)
10. Regulation (EU) 2016/679 (GDPR), Articles 4(1), 4(7) and 32. https://eur-lex.europa.eu/eli/reg/2016/679/oj/eng (accessed 2026-09-12)
11. California Civil Code §1798.100(e) (reasonable security) and §1798.145(m) (employee-data exemption inoperative from 1 Jan 2023), https://leginfo.legislature.ca.gov/faces/codes_displaySection.xhtml?lawCode=CIV&sectionNum=1798.100; California Privacy Protection Agency, "Frequently Asked Questions", on the expired employee-data exemption, https://cppa.ca.gov/faq (accessed 2026-09-12)
12. European Data Protection Board, "Guidelines 02/2025 on processing of personal data through blockchain technologies", version 2.0, adopted 7 Jul 2026. https://www.edpb.europa.eu/documents/guideline/guidelines-022025-on-processing-of-personal-data-through-blockchain_en (accessed 2026-09-12)
13. Toku, "Toku Expands Stablecoin Payments on Polygon to 100+ Countries, Processing More Than $1 Billion in Annual Volume", 16 Jan 2026. https://www.toku.com/resources/toku-expands-stablecoin-payroll-on-polygon-to-100-countries-1b-annual-volume (accessed 2026-09-12)
14. Circle, "USDC Now Available Natively on Base", 5 Sep 2023. https://www.circle.com/blog/usdc-now-available-natively-on-base (accessed 2026-09-12)

15. Circle, "USDC Terms", section on blocked addresses and forfeited funds. https://www.circle.com/legal/usdc-terms (accessed 2026-09-12)
16. Toku, "How Toku Runs Fully Private Stablecoin Payroll on Aleo and USAD", 2026. https://www.toku.com/resources/how-toku-runs-fully-private-stablecoin-payroll-on-aleo-and-usad; Aleo, "Private Stablecoins for Payroll", https://aleo.org/payroll/ (accessed 2026-09-12)
17. Zama, "Confidential Onchain Payroll", https://www.zama.org/solutions/confidential-onchain-payroll; "Confidential USDC (cUSDC)", https://www.zama.org/confidential-tokens/cusdc; Zama Protocol documentation, "Overview" (Key Management Service as a threshold MPC network), https://docs.zama.org/protocol/protocol/overview (accessed 2026-09-12)
18. Hinkal, "Best Privacy Rails for Crypto Payroll Payments", https://hinkal.io/blog/best-privacy-rails-crypto-payroll-payments; "Hinkal vs Zama vs Aleo", 9 Jun 2026, https://hinkal.io/blog/hinkal-vs-zama-vs-aleo (vendor-authored; accessed 2026-09-12)
19. Greenberg, A., García, E., Croubois, H., Ben Amor, G., Danjou, C., Turk, J. A., Davis, S., Pasquier, N., "ERC-7984: Confidential Fungible Token" (transfer functions and events carry plaintext addresses; amounts are confidential handles). https://eips.ethereum.org/EIPS/eip-7984 (accessed 2026-09-12)
