COSIC SEMINAR · KU LEUVEN · ONLINE

Applied Cryptography for Privacy in Ethereum

Usecases, constraints, and requirements

Ali Atiia

Private Reads team · Ethereum Foundation

`privreads.ethereum.foundation`

THIS TALK

Three parts

1 · Ethereum

What it is, the three pillars its R&D pushes on, and who uses it today.

2 · R&D map

Ongoing work through an applied-crypto lens: post-quantum and scaling-by-SNARK.

3 · More about privacy

In-protocol privacy EIPs, the access layer, and private reads: PIR for Ethereum state.

01

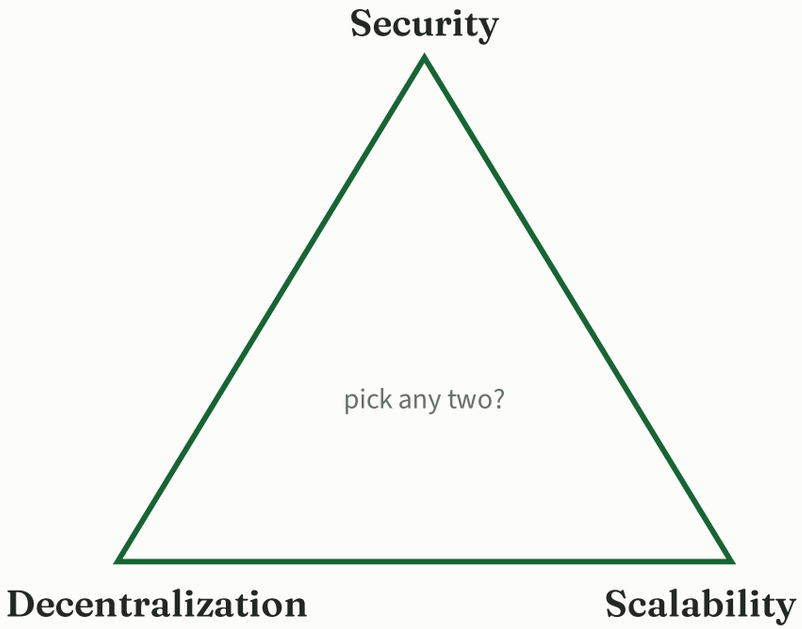
Overview of Ethereum R&D

A proof-of-stake blockchain; the three pillars (security, decentralization, scalability).

Ethereum is an applied-cryptography R&D program

- Cutting-edge R&D has gone, and still goes, into:
  - **breaking the trade-off** between **the three pillars every blockchain must balance**: security, decentralization, scalability;
  - hardening the guarantees of **permissionless, censorship-resistant access** to the protocol, in a **privacy-preserving** manner.

Three pillars, one tension



- **Security**: slashing & recovery, faster finality (single-slot finality R&D), censorship-resistance (inclusion lists), post-quantum track.
- **Decentralization**: stateless validation (verify blocks from witnesses, not a full state copy), client diversity, consumer-hardware staking.
- **Scalability**: rollups + blob data (EIP-4844; PeerDAS sampling live since the Fusaka fork, Dec 2025), zkVM-proven execution, L1 gas-limit growth.

Cryptography, namely SNARKs and data-availability sampling, is the lever that relaxes the trilemma instead of trading one pillar for another.

The one last problem to solve before the trilemma is fully overcome: **incentivised hosting and sharing of partial state**, for the future where the Ethereum state becomes extremely large.

> TLDR: Ethereum blockchain is: BFT consensus + VM + merklized state and logs.

Proof of stake

- Validators bond ETH (32 minimum) and run a node; misbehavior (equivocation) is **slashed**: security is economic, not computational.
- Time is divided into **12 s slots**; one validator proposes a block, a committee **attests**; every validator votes once per 32-slot epoch.
- **Finality**: a ⅔-supermajority checkpoint vote (Casper FFG) makes blocks irreversible unless ≥⅓ of stake is destroyed, a unique balance of safety/liveness [2].
- Feasible only because of **signature aggregation**: ~1M attestations per epoch compress to a few signatures per block.

Adoption

~$50B
DeFi value locked on Ethereum L1, ~54% of all-chain DeFi (DefiLlama)

~$160B
stablecoin supply on Ethereum, of ~$320B globally (DefiLlama)

~1.1M
active validators; ~36M ETH staked, ~29% of supply (beaconcha.in)

5 + 6
independent execution + consensus client implementations, in 4+ languages (clientdiversity.org)

~32k
active open-source developers in the ecosystem (Electric Capital, 2025)

~73
live rollups (L2s) securing >$48B (L2BEAT)

Example applications

APPLICATION	FOR A CRYPTOGRAPHER, IT IS…
Stablecoins	digital dollars as on-chain tokens, the dominant payment use; settlement without correspondent banking.
DeFi	exchanges, lending, derivatives as open smart contracts: financial logic with public, auditable state.
Prediction markets	markets on real-world events with on-chain settlement (e.g. Polymarket).
ENS	the on-chain analogue of DNS: human-readable names → addresses, keys, records.
Collectibles / NFTs	a global ownership registry for digital (and tokenized physical) objects.
Governance / DAOs	on-chain voting and treasury management for internet-native organizations.

02

Ethereum R&D, through an applied-crypto lens

The deployed cryptography, the post-quantum migration,
scaling by SNARKs, and a map of the privacy work.

Upgrading to post-quantum crypto

The cryptography running Ethereum today

PRIMITIVE	WHERE IT RUNS	QUANTUM
ECDSA / secp256k1	every user transaction (account signatures)	**Shor-broken**
BLS12-381 aggregate sigs	consensus attestations (~1M validators/epoch)	**Shor-broken**
KZG commitments	blob data availability for rollups (EIP-4844)	**Shor-broken**
Keccak-256	state trie, addresses, content addressing	fine (Grover only)
SNARKs	rollup validity proofs	zkVM STARK proofs are PQS, but Groth16 proofs that wrap them (for cheap on-chain verifiability) are not

All **non-PQS** components must be upgraded away: **on the live system, with no maintenance window.**

Lean Ethereum: three herculean replacements

Consensus SIGNATURES	Data COMMITMENTS	Execution ACCOUNTS
~~BLS aggregate signatures~~	~~KZG polynomial commitments~~	~~ECDSA account signatures~~
↓	↓	↓
Hash-based signatures leading candidate: XMSS, aggregated by SNARKs	**Hash-based commitments** FRI-style, coding-theoretic	**PQ signatures** opted in per account ("account abstraction")

Each one: live migration, ~1M participants, zero downtime. The hardest is consensus, where **BLS aggregation has no cheap post-quantum analogue: everything reduces to hash functions, plus SNARKs over hashes.**

leanVM, and ~$2M of open invitations

- **leanVM**: a minimal zkVM for SNARK-based signature aggregation, built on **WHIR** (hash-based proximity proofs) + **Poseidon2** over the 31-bit **KoalaBear** field.
- Leading signature candidate: **XMSS**, with public key ~52 B (≈BLS-sized) but signatures ~3.1 kB → **SNARK aggregation is mandatory**, not optional.
- Both performance and safety hinge on **SNARK-friendly hashes** and on **proximity testing** for Reed–Solomon codes.

ethresear.ch/t/exploring-the-design-space-for..

PQ key-registry design space

Proximity Prize · $1M

two challenges on Reed–Solomon proximity gaps (mutual correlated agreement; list decoding), judged by Boneh, Fenzi, Arnon. `proximityprize.org`

Poseidon Initiative · ~$1.25M

cryptanalysis programs for Poseidon/Poseidon2, incl. a $992k collision prize (to 2029) and yearly bounties. `poseidon-initiative.info`

Scaling

Scaling by verifying, not re-executing

- Today every full node **re-executes every transaction**: the gas limit is bounded by the slowest acceptable node.
- **zkVMs** change the model: the block builder proves the block's execution with a SNARK; everyone else verifies it and accepts the block without having to re-execute.
- ethproofs.org tracks the race: independent zkVM teams proving real mainnet blocks: currently **~94% of tracked proofs land in ≤10 s**, at **~$0.01–0.03 per block proof**.
- Target: **real-time proving**, the proof ready within the 12 s slot, which raises the gas limit and lets phones verify full blocks; the roadmap ends at **enshrined L1 proofs**.

ethproofs.org

03

More about privacy

Protocol-amending EIPs, access-layer privacy, and private reads: PIR over Ethereum state.

MORE ABOUT PRIVACY

The privacy R&D map

In-protocol

EIPs amending L1 to make private applications cheaper and native: confidential transfers, stealth addresses, privacy-friendly opcodes & precompiles.

Extra-protocol

privacy between users and infrastructure: network-level transport (Tor-style), private RPC reads (PIR), encrypted mempools.

IRL bridges

making web2 facts verifiable on-chain without revealing them: zk-TLS (prove a TLS session's contents: TLSNotary), zk-ID (prove personhood / attributes: ZKPassport).

I will zoom into the first two.

MORE ABOUT PRIVACY

Everything on Ethereum is public by default

- Everything on-chain is **public, forever**: balances, transfers, votes, names, counterparties; by design, that is what makes it verifiable.
- And trustless access requires a **full node** (>1 TB NVMe, days to sync), so almost everyone reads the chain **through someone else's server**.
- So even *reading* public data leaks your **assets, affiliations, and counterparties** from query patterns alone; no cryptography is broken.

Transparency is the feature. Surveillance is the side-effect.

MORE ABOUT PRIVACY

Two surfaces of privacy leakage

Write privacy (on-chain)

What you **put on-chain**: transfers, balances, votes. Adversary: everyone, forever.

Tools: ZK proof systems, stealth addresses, privacy pools, confidential transfers.

Read privacy (off-chain)

What you **ask about the chain**: queries to RPC infrastructure. Adversary: your provider (and whoever watches it).

Tools: PIR, ORAM-style techniques, anonymous transport, query unlinkability.

Privacy leakage from reads: a curious or malicious server can profile you just by tracking what you *read*, undermining privacy measures you worked hard to follow elsewhere. Shielding your transactions doesn't help if the read pattern alone tells the same story.

ethereum-magicians.org/t/a-maximally-simple..

Privacy Roadmap

MORE ABOUT PRIVACY · IN-PROTOCOL

Recent proposals to improve the ergonomics and lower the cost of privacy transactions

all drafts · June 2026

EIP-8141

Frame transactions

The foundation: native account abstraction, so **one shared sender can act for many users**.

EIP-8250

Keyed nonces

Transactions flowing through a shared pool become unlinkable, and can be processed in parallel.

EIP-8272

Recent roots

Lowers the cost of privacy transactions.

- **Already standard:** stealth addresses, via ERC-5564 (final) + ERC-6538 registry: non-interactive one-time addresses with view tags.
- **Censorship-resistance as a complement:** FOCIL (EIP-7805, draft) inclusion lists guarantee privacy transactions can't be quietly excluded.
- Privacy has become a major priority for the Ethereum Foundation and the ecosystem at large.

Privacy at the access layer

the periphery of the protocol, where you query it

The access layer: privacy of traffic and query contents from remote state providers

- Provider learns: your **IP**, **all your addresses in one breath** (linkage), the assets and dapps you care about, and **when** you check them.
- Concentration makes it worse: most wallet traffic defaults to a handful of providers.

This layer is where my team (**Private Reads**, EF) works.

TorJS: network-level privacy where wallets live

- Wallets are browser extensions and web apps; they can't ship a Tor daemon. **TorJS** brings a Tor client into the JS/browser environment, so RPC traffic can be onion-routed from inside the wallet.
- What it fixes: the **IP → identity** link. The provider no longer knows *where* the query came from.
- What it cannot fix: the **query content**: addresses inside the query still link to each other and profile the (now pseudonymous) user.

Anonymous transport is necessary but not sufficient: content-level protection needs PIR.

privreads.ethereum.foundation/docs/torjs

TorJS docs

An abstract access layer: anon-rpc

- Today each wallet hard-wires its own provider integrations; privacy tech would need to be re-integrated **wallet by wallet**.
- **anon-rpc**: a common abstraction over how clients reach chain data, so transports (Tor, mixnets) and query protections (PIR) **slot in beneath every wallet at once**.
- Goal: make the private path the **default path**, not a power-user opt-in.

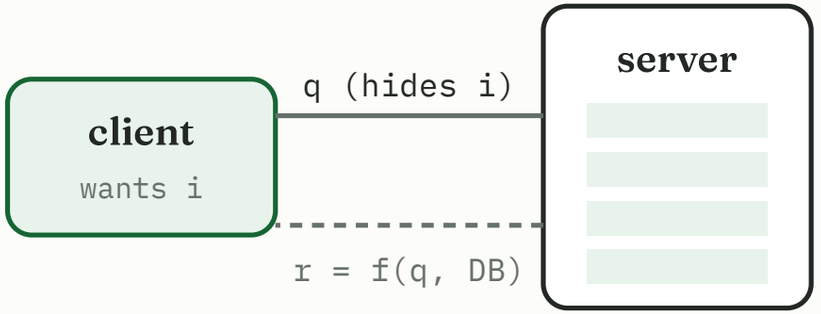
> ongoing work

privreads.ethereum.foundation/feed/anon-rpc

anon-rpc design post

Private Information Retrieval

- Database of N records; client wants record i; **the server must learn nothing about** i.
- Trivial scheme: download everything, at communication $O(N)$. PIR asks for **sublinear communication**.
- Two families by security: **IT-PIR** (information-theoretic, needs 2+ non-colluding servers) and **cPIR** (computational security; enables a single server, homomorphic-encryption based today).
- The catch: without preprocessing, every server must **touch every record per query**, $\Omega(N)$ work; a skipped record reveals it is not i. Preprocessing lifts this, but it also comes with its own issues.



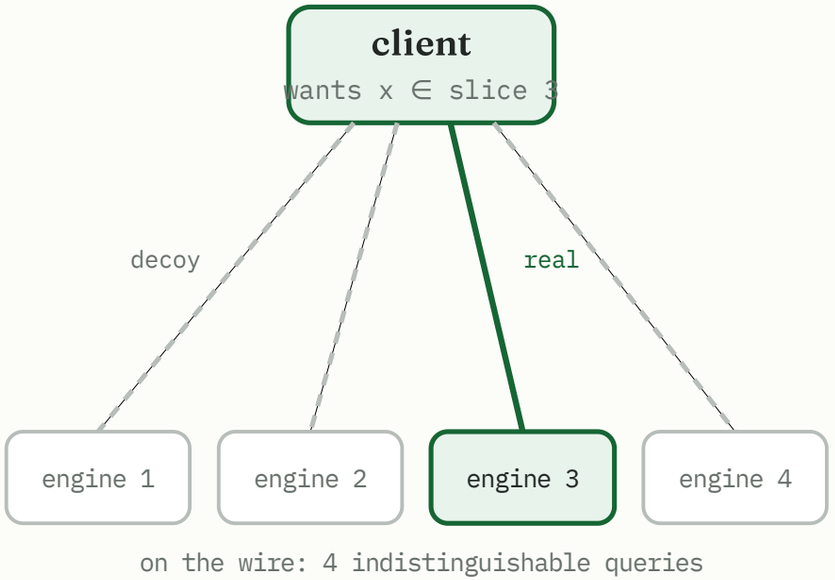
The taxonomy, against Ethereum's constraints

CLASS	STRENGTH	PAIN POINT AT CHAIN SCALE
IT multi-server	fast (XOR), no lattices	needs non-colluding operators, hard to certify
Single-server cPIR (RLWE)	one untrusted server	heavy server compute, big queries
Stateful / preprocessing	sublinear *online* time via client hints	hints break when the DB mutates: **Ethereum's state changes every 12 s**
Doubly-efficient PIR	sublinear server time, no per-client state	practical constructions: open problem

github.com/0xalizk/PIR-Eng-Notes

Sharded PIR for the Ethereum state

- **One PIR engine over everything is impractical**: too intractable for double-stateless $O(N)$ schemes (e.g. exceeds GPU memory), and too volatile for preprocessed schemes, which need to refresh precomputed "hint" structures after mutations.
- **So: slice by update & access profile**, from curated hot data (1–10 GB) to full archival state (2–20 TB), and pair each slice with the **PIR scheme that fits it**.
- **Privacy is restored by decoys:** the client sends its genuine query to one engine and *real* decoy queries to all the others, in parallel.



Observer's view = monolithic PIR over the whole state. Performance = per-slice optimized: the genuine answer arrives at the small slice's latency, not the monolith's.

> Decoys are real queries: each engine sees a uniform access either way. Subtleties (timing side-channels → wait-for-all / m-of-k; all-data-or-nothing coverage).

Doubly-stateless PIR, tailored to GPUs

- **Doubly-stateless**: no per-client state on the server *and* no client-side hint for updates to invalidate: any client queries cold; the 12 s mutation costs nothing per client.
- Why insist on it: users query the state from all sorts of contexts (wallets, dApps, a website they are just passing through), so statelessness makes the PIR solution applicable to **everyone**, requiring no *a priori* storage or preprocessing on their part.
- **InsPIRe** (current focus):
  - single-server lattice PIR with **silent preprocessing** (server-side only, no hint download)
  - vs. YPIR: ~50% smaller queries, 5× smaller keys, up to 25% more throughput
    - ~36 ms latency per query, 390 KB communication, ~6.5 s setup, on a 16 GB DB
- The price is raw server compute, a **memory-bandwidth-bound linear scan, shaped for GPUs**: our GPU port of InsPIRe is underway, with promising early numbers.

> The paper's benchmarks are CPU; the GPU figures are our own in-progress, unpublished work.

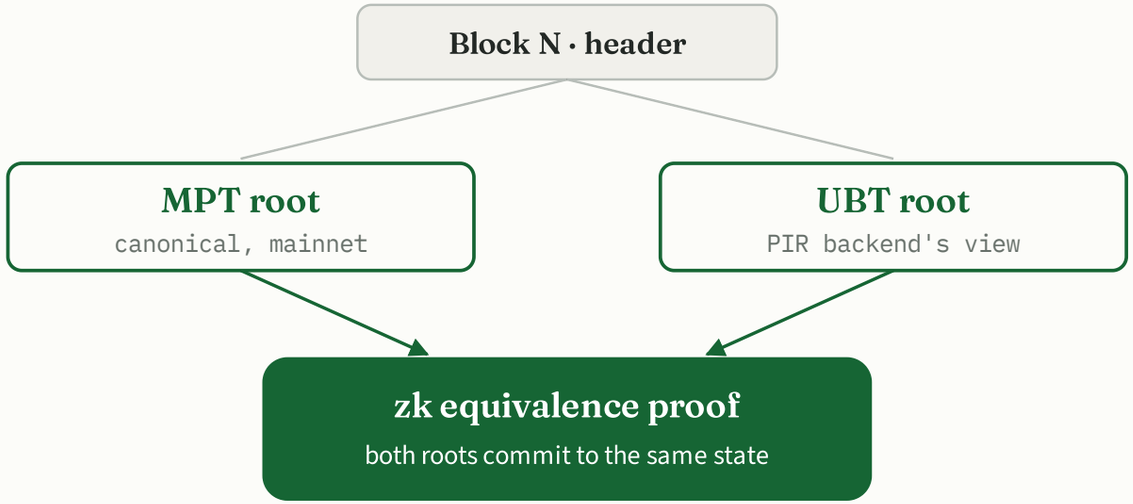
Open problems

- **Doubly-efficient PIR**: drive online server work sublinear, touch strictly fewer bits than the database per query, with communication below the database size, so answering beats just downloading the state. Today this needs heavy preprocessing or non-standard assumptions.
- **Hint delegation under a frequently mutating database**: client-stateful schemes precompute a hint that a 12 s state change can invalidate; cheap incremental repair, or *delegating* the preprocessing itself (FHE-based, round-optimal, cf. ThorPIR), is barely explored.
- **Hardware acceleration**: doubly-stateless lattice PIR is a memory-bandwidth-bound linear scan; GPU/FPGA is the lever (current GPU state of the art: GPIR, 2026). The bar is hundreds of GB at thousands of queries per second, sub-100 ms.
- **Provable security & batching**: 128-bit security under *standard* assumptions, not heuristics; plus many-client batching and anonymity-set-aware sharding.

A PIR solution for the entire Ethereum state (300 GB+ hot, 2 TB+ archival) would be the largest demonstration of PIR to date. The few existing PIR deployments serve comparatively tiny databases.

Using a PIR-friendlier state trie

Scale to serve: ~10^8 accounts, ~10^9 storage slots, **300 GB+** hot state, mutating **every 12 s**, orders of magnitude beyond most PIR benchmarks.



$$\forall\,(k,v):\ (k,v)\in \mathrm{MPT} \iff (k,v)\in \mathrm{UBT}$$

- The PIR backend can serve the **UBT** (flat, index-addressable, one SNARK-friendly hash) while mainnet keeps the **MPT**.
- A per-block **zk proof binds the two roots** to the same state, so private reads get the PIR-friendly structure *before* it ships on-chain.
- It pays off: a read under PIR costs **~48×** on the MPT versus **~9×** on the binary tree (our analysis).

privreads.ethereum.foundation/workstreams/ubt/

> EIP-7864 is draft, not scheduled; the hash choice is open (BLAKE3 in the current draft; Poseidon2 and Keccak candidates). A geth binary-trie prototype today is ~1.7× slower on reads and ~2.5× on writes than the MPT (CPerezz benchmarks), optimization ongoing. Verkle, its EC-based predecessor, was de-prioritized largely for post-quantum reasons.

OUTRO

Summary of open problems in Ethereum R&D generally

PQ

Post-quantum signatures

Aggregation-friendly hash-based schemes, SNARK aggregation at ~1M-validator scale, the PQ key registry.

CRYPTANALYSIS

SNARK-friendly hash cryptanalysis

Poseidon2 under real-money bounties (the $992k collision prize), Reed-Solomon proximity gaps (the $1M Proximity Prize).

PIR

PIR at chain scale

Doubly-stateless schemes, GPU/FPGA acceleration, preprocessing under mutation.

TLS

MPC-TLS / zk-TLS

Turning designated-verifier proofs into publicly verifiable (public-coin) proofs, making proofs about web2 consumable on-chain.

THANK YOU

Questions, and pointers

privreads.ethereum.foundation

the team

ethresear.ch

Ethereum Research

ethereum-magicians.org

Technical Forum