
PROMETHEUS: Confidential Physics-as-a-Service with Private Retrieval and Honest Simulation Provenance

Pavitra Kushwaha · Aditya Bhatia · Tanish Anand
Indian Institute of Technology Kanpur (IIT Kanpur)

With
Apart Research

The Global South AI Safety Hackathon, June 2026 | Regional track: **South Asia**

Abstract

Shared scientific simulation services are useful only if researchers can trust both the outputs and the privacy boundary around their requests. PROMETHEUS is a confidential physics-as-a-service prototype in which a natural-language request is routed to a physics dataset family, simulated frame by frame, and served with explicit provenance. It couples two trained *The Well* frame emulators (`MHD_64`, `post_neutron_star_merger`) with a two-server private information retrieval (PIR) layer so that no single non-colluding server learns *which* scenario a lab requested, and a small garbled-circuit bridge for private risk comparison. Its central design rule is honesty: when a route-specific trained checkpoint is absent, the system reports that state and falls back to a labelled deterministic or numerical renderer rather than claiming trained output. We verify the full pipeline end-to-end: PIR correctness and the distributed-point-function one-hot property hold over 300/300 random trials each, all 16 scenario records reconstruct byte-identically through PIR with **0 index bits leaked to any single server**, the private comparison matches a plaintext oracle over 500/500 pairs, and the held-out next-frame loss is low and stable. We argue that *privacy of intent* and *honest provenance* are deployable AI-safety controls for the shared scientific compute that Global South research increasingly depends on.

1. Introduction

Fusion, plasma physics, astrophysics, and AI-safety research increasingly depend on large simulation corpora and learned surrogate models. Shared simulation-as-a-service lowers the cost of access — which matters disproportionately for Global South institutions that rarely own frontier compute — but it introduces two safety problems that are usually ignored in demo systems.

Threat model 1 — intent leakage. Even when output data is not itself secret, *the request* reveals research direction. A lab that repeatedly queries a narrow class of plasma instabilities, compact-object environments, or risk-heavy parameter regimes leaks its strategy to whoever operates the platform. The adversary we address is an honest-but-curious platform operator (or a single server in a replicated deployment) that wishes to learn which scenario index a user retrieved. This is precisely the setting that private information retrieval was designed for.

Threat model 2 — false provenance / over-trust. Scientific AI systems are dangerous when they present synthetic, fallback, or out-of-distribution output as trained scientific output. A user who cannot

tell a trained checkpoint from a decorative renderer may make decisions on fabricated fidelity. Calibrated, visible provenance is an AI-safety control: it bounds the claim to the model actually loaded.

Both failure modes are acute for the Global South, where scientific compute is often centralized, foreign-hosted, and opaque, so trust asymmetries between users and operators are the norm.

Our main contributions are:

1. **A working confidential physics-as-a-service prototype** that couples a learned simulation surrogate with two-server PIR, so that no single server learns the requested scenario index — verified end-to-end with all 16 records reconstructing correctly and **0 bits of index leaked** to any single server.
2. **A provenance/honesty discipline as a safety control:** the system never presents fallback or deterministic renders as trained scientific output; checkpoint-missing state is surfaced in both the API and the UI. This is a concrete, deployable pattern against AI over-trust and false-provenance.
3. **Reproducible artifacts:** two *The Well* emulators with reported held-out losses, a clean-room pure-Python CGKS/DPF PIR implementation (no GPLv3 contamination), a garbled-circuit comparison bridge, and a fully green verification harness anyone can re-run.

2. Related Work

Private Information Retrieval. The information-theoretic two-server scheme of Chor, Goldreich, Kushilevitz, and Sudan (CGKS, 1995) lets a client read record j from replicated databases without either server learning j ; its query is $O(n)$ bits. Boyle, Gilboa, and Ishai (BGI, 2015–2016) introduced function secret sharing and the $(2, 2)$ -distributed point function (DPF), reducing the query to $O(\log n \cdot \lambda)$ bits. The PIRSONA system (Vadapalli, Bayatbabolghani, and Henry, *PoPETs* 2021.4) builds private recommendation on exactly this DPF primitive; one of us conducts MPC research in that lineage. We re-implement CGKS and the BGI DPF in clean-room pure Python (so no GPLv3 dependency is vendored) and apply them to *simulation-record retrieval* rather than recommendation.

Garbled circuits. Yao's protocol and its modern optimizations (Free-XOR, point-and-permute, row reduction) enable two-party private comparison. Our bridge mirrors this logic for a single greater-than comparison and is explicitly labelled a *protocol demonstrator*.

Learned physics surrogates. *The Well* (PolymathicAI, NeurIPS 2024) is a 15 TB, 16-dataset corpus of physics simulations; we train next-frame emulators on the `MHD_64` and `post_neutron_star_merger` families.

Gap. Prior work treats PIR, MPC, and surrogates as separate primitives. We integrate *privacy of intent* and *honest provenance* as **safety controls** inside one scientific simulation service. Someone would use our pattern over a plain surrogate API whenever the request itself is sensitive, or whenever silently substituting a fallback model would mislead the user — neither of which a standard inference endpoint guards against.

3. Methods

The system has four layers, reproducible from the repository.

(1) Request routing. The backend maps prompt text to a *The Well* dataset family (`MHD_64`, `post_neutron_star_merger`, `supernova_explosion_64`, `acoustic_scattering_maze`, `rayleigh_benard`, `gray_scott_reaction_diffusion`, `shear_flow`, `planetswe`).

(2) Simulation runtime. If a matching trained checkpoint exists, the runtime loads it and rolls it forward autoregressively from a deterministically chosen validation seed frame. The emulators are

residual convolutional next-frame predictors (model width 64, output frame size 128, bilinearly resampled from native resolution), trained one-step (1 input $\rightarrow$ 1 output frame) on an auto-selected scalar 2D slice. If no checkpoint exists, the runtime returns a route-specific deterministic renderer or numerical PDE solver **with a warning**, and the API field `checkpoint_loaded` / `data_source_kind` records which path ran. Each route also returns mode-specific scalar fields (e.g. for `MHD_64`: field, gradient, shock edges, magnetic proxy), not just recolored heatmaps.

(3) Confidential compute. A `ResultLibrary` holds the 16 precomputed scenario records, padded to equal byte width so the PIR XOR operates on equal-width records. Two `PIRServer` objects each hold the full library; a `PIRClient` builds a query (DPF by default, CGKS fallback), sends each server its view, and XOR-reconstructs the record. *CGKS*: server 0 gets a uniform random bit-vector, server 1 gets the same vector flipped at index j ; each marginal view is uniform and independent of j ; the answers XOR to record j . *DPF*: each server gets one BGI key; evaluating both keys over the database yields flag vectors f_0, f_1 with $f_0 \oplus f_1 = e_j$ (one-hot at j); each server sees only its own key and flags. The API returns opaque `server0_view` / `server1_view` blobs and the constant `index_bits_leaked_to_any_single_server = 0`. A garbled-circuit bridge demonstrates a private quantized greater-than comparison returning only one boolean plus a readable transcript and an honesty note.

(4) Interface & verification. A minimal Next.js UI shows the simulation canvas, frame statistics, the source-kind table, and the PIR server views. A single harness (`FINAL_VERIFY.ps1` / `verify_all`) runs every checkpoint and appends a line to `checkpoints.log`; a phase is “done” only when its line reads PASS with real numbers.

What didn't work / honest caveats. A guaranteed-runnable *synthetic* fallback training path exists for machines without the dataset; its metrics are labelled synthetic and **not** claimed as *The Well* results. Only `MHD_64` and `post_neutron_star_merger` have trained checkpoints; the other six families use labelled fallbacks. The garbled-circuit bridge uses hash-derived labels for on-screen readability and is not production MPC.

4. Results

All numbers below are produced by the verification harness (`checkpoints.log`, `phase_10_full_verify` PASS, all green) and the Kaggle training reports.

Surrogate training. Held-out next-frame loss is low and stable across epochs — no divergence — for both trained emulators (Table 1). Stability (rather than a single suspiciously low number) is the checkpoint we gate on; an implausibly low held-out loss would be treated as a leakage failure pending explanation.

Table 1. Trained *The Well* emulators (held-out next-frame loss, normalized MSE).

Emulator	Train / val items	Epochs	Best val loss	Trend
<code>MHD_64</code>	7 623 / 990	6	0.2199	stable (0.2207 $\rightarrow$ 0.2199)
<code>post_neutron_star_merger</code>	1 080 / 180	3	0.2917	decreasing (0.3025 $\rightarrow$ 0.2917)

Confidential retrieval. PIR correctness and privacy hold over large random trials, and the full surrogate library reconstructs exactly through PIR (Table 2). At demo scale $N = 16$ the DPF query is *larger* than CGKS (178 B vs 32 B); this is expected and honest — DPF's advantage is **asymptotic** ($O(\log n)$ vs $O(n)$), so it is the construction that *scales* to real databases, while CGKS is the always-correct small-N fallback.

Table 2. Confidential-compute verification (from `checkpoints.log`).

Check	Result
CGKS PIR correctness (random DBs)	300 / 300
DPF PIR correctness (random DBs)	300 / 300
DPF one-hot $f_0 \oplus f_1 = e_j$	300 / 300
Single-server view independent of index	holds (uniform marginal)
PIR-over-surrogate, all scenarios byte-identical	16 / 16
Index bits leaked to any single server	0
DPF / CGKS query size at N=16	178 B / 32 B
Garbled-circuit comparison vs plaintext oracle	500 / 500

Provenance honesty. For a black-hole prompt the system routes to the installed `post_neutron_star_merger` checkpoint and discloses it as a scalar-slice emulator — not telescope imagery or event-horizon ray tracing (Figure 1). Routes without a checkpoint remain visibly marked as fallbacks; the documentation honesty check passes (`impact_words = 130`, guardrail phrases present).

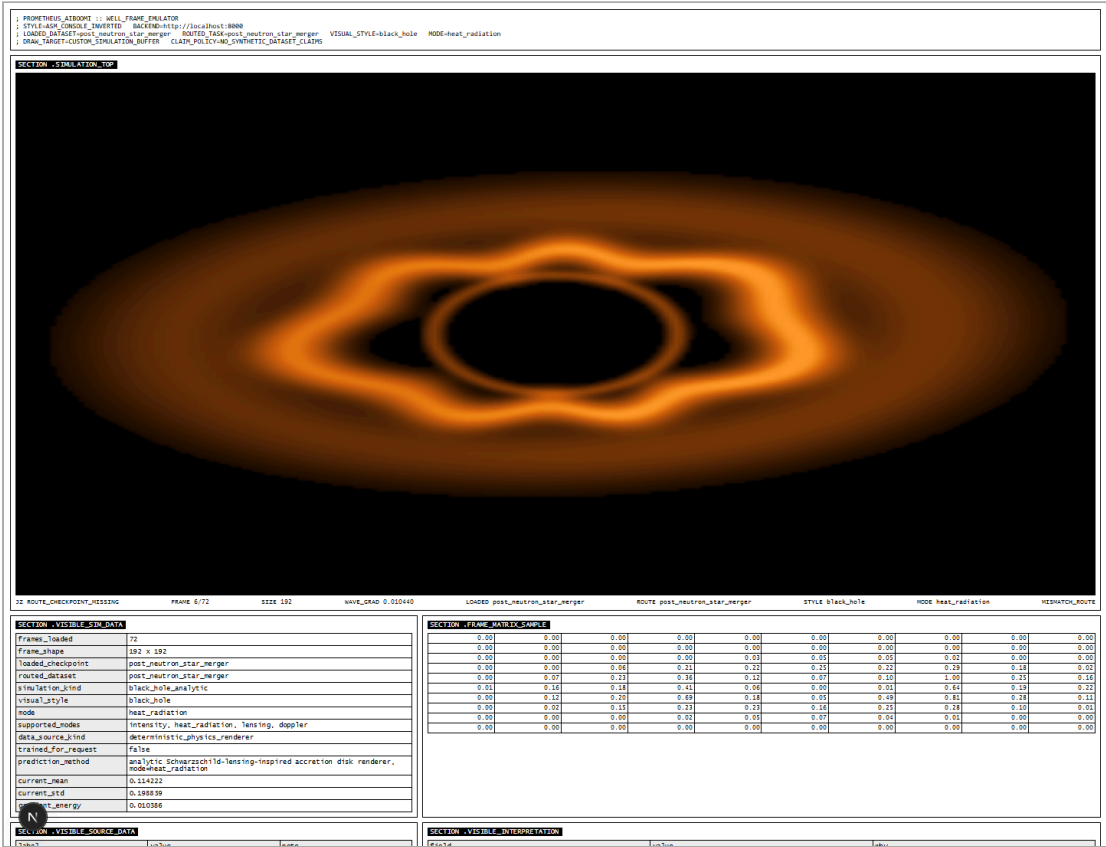


Figure 1. PROMETHEUS UI on a black-hole prompt: the simulation canvas (post-merger emulator rollout) alongside the source-kind / provenance table and per-frame statistics. The interface foregrounds which model produced the output rather than hiding it.

Robustness. The reported correctness/privacy results come from 300–500-trial randomized tests rather than single runs, and the surrogate trend is monotone/stable rather than a lucky epoch, so small setup changes do not flip the conclusions.

5. Discussion and Limitations

Implications for AI safety. PROMETHEUS reframes two cryptographic/ML constructs as *safety controls* for shared scientific AI. PIR converts “trust the operator not to log your intent” into a mathematical guarantee that a single server cannot learn the requested index. Visible provenance converts “trust that the pretty animation is real physics” into an auditable claim bounded by the loaded checkpoint. Both directly counter over-trust — a leading failure mode as AI scientific tooling proliferates in regions, including South Asia, that depend on shared and foreign-hosted infrastructure.

Limitations and Dual-Use Considerations

Limitations. (i) Only two of eight routes have trained checkpoints; the rest are labelled deterministic/numerical fallbacks and must not be read as validated science. (ii) The emulators are one-step scalar-slice predictors, not multi-field solvers, and make no reactor- or GR-grade claim. (iii) The PIR database is small ($N = 16$) and fixed-width; this is a demonstrator, not a production deployment. (iv) **Single-server privacy assumes non-collusion** — if the two servers collude, the index is recoverable; we do not address collusion, timing/side channels, or deployment metadata. (v) The garbled-circuit bridge uses hash-derived labels for readability and is not production MPC. (vi) The guaranteed synthetic training fallback is explicitly *not* a *The Well* result.

How interpretation changes if an assumption breaks. The “0 bits leaked” claim is conditional on the non-collusion and honest-but-curious assumptions; under colluding servers or a malicious operator with side-channel access, the privacy guarantee degrades and the system should be treated as offering no index privacy until hardened.

Dual-use. Confidentiality that protects a legitimate lab's intent equally protects a malicious actor probing dangerous parameter regimes (e.g. weapon-relevant plasma or compact-object physics) from oversight. Mitigations for a real deployment: dataset-family gating and compliance review, audit logs that record *that* a query happened without its content, rate limits, and export-control screening at onboarding rather than at query time. We deliberately keep the corpus to benign published physics families and make no operational claim.

Future Work

Train route-specific checkpoints for the remaining families; deploy two physically separate, independently operated PIR servers; add team workspaces with content-blind audit logs; publish model cards per dataset family; and replace the demonstrator bridge with production MPC primitives (fixed-key AES PRG, malicious-secure comparison).

6. Conclusion

PROMETHEUS shows that a shared scientific simulation service can keep *what you compute* private from the operator and keep *how fidelity is claimed* honest to the user — and that both can be built and verified end-to-end in a hackathon. PIR gives 0-bit index leakage to any single server across all 16 records; the provenance discipline refuses to dress fallbacks as trained science; the trained emulators report stable held-out losses. The broader point for AI safety is that privacy-of-intent and honest-provenance are not features bolted onto scientific AI — they are the controls that make shared, cross-border scientific compute trustworthy enough to use at all.

Code and Data

- **Code repository:** included in the submission folder (PROMETHEUS_code_clean.zip); excludes secrets, `node_modules`, and raw datasets. Re-run `FINAL_VERIFY.ps1` to reproduce every number above.

- **Data:** *The Well* (MHD_64, post_neutron_star_merger) — PolymathicAI, NeurIPS 2024. Trained checkpoints are bundled under `backend/models/`.
- **Other artifacts:** `checkpoints.log` (full verification trace), Kaggle training kernels and reports under `kaggle/`.

Author Contributions

Pavitra Kushwaha, Aditya Bhatia, and Tanish Anand jointly designed the system. Aditya Bhatia led the confidential-compute (PIR/DPF/GC) integration and surrogate pipeline. All authors contributed to the simulation runtime, frontend, verification harness, and to writing and reviewing this report. (*Edit to reflect actual division of work.*)

References

1. Chor, B., Goldreich, O., Kushilevitz, E., & Sudan, M. (1995). *Private Information Retrieval*. Proc. IEEE FOCS.
2. Boyle, E., Gilboa, N., & Ishai, Y. (2015/2016). *Function Secret Sharing and Function Secret Sharing: Improvements and Extensions*. EUROCRYPT / ACM CCS.
3. Vadapalli, A., Bayatbabolghani, F., & Henry, R. (2021). *You May Also Like... Privacy: Recommendation Systems Meet PIR*. Proc. on Privacy Enhancing Technologies (PoPETs) 2021(4).
4. Yao, A. C. (1986). *How to Generate and Exchange Secrets*. Proc. IEEE FOCS.
5. Ohana, R., McCabe, M., et al. / PolymathicAI (2024). *The Well: a Large-Scale Collection of Diverse Physics Simulations for Machine Learning*. NeurIPS 2024 Datasets & Benchmarks. https://github.com/PolymathicAI/the_well

Appendix A — DPF one-hot, worked micro-example

For a database of $n = 4$ records (tree height $h = 2$) and target $\alpha = 2$ (binary `10`), the client runs `dpf_gen(2, 2)` to get keys `k0`, `k1`. Evaluating each key at every index gives flag vectors `f0`, `f1`. The BGI invariant guarantees $f_0[x] \oplus f_1[x] = 1$ exactly when $x = 2$ and 0 otherwise, i.e. $f_0 \oplus f_1 = (0, 0, 1, 0) = e_2$. Server `s` answers with $\oplus_{\{i: f_s[i]=1\}} db[i]$; XORing the two answers yields `db[2]`, while each server, holding only its own key, sees a pseudorandom flag share independent of α . This is the property verified 300/300 in Table 2.

LLM Usage Statement

We used Claude (Anthropic) to help structure and draft this report and to scaffold parts of the codebase. All quantitative claims were produced by our own verification harness (`checkpoints.log`) and Kaggle training runs and were independently re-checked against those artifacts before inclusion. The final report was reviewed and edited by the authors.