A Deterministic, Verifiable LLM Deployed on Fogo

Neutral intelligence as blockchain infrastructure

THESIS

Blockchains made computation neutral. Baranos AI extends that property to inference: a committed model, committed inputs, deterministic execution rules, and a result that can be independently reproduced, challenged, and resolved.

- **DETERMINISTIC**
  Same committed job → same canonical result
- **CHALLENGEABLE**
  Incorrect execution can be disputed and resolved
- **COMPOSABLE**
  Results can become state consumed by smart contracts

Version 1.6 / September 2026

Current deployment: Fogo

Status note: deployment, model-performance, cost, challenge-resolution, and cross-chain capability claims should be supported by reproducible public benchmarks before being treated as independently verified production guarantees.

Abstract

Baranos AI is a verifiable onchain inference system deployed on Fogo. Its normal operating path uses optimistic verification: computationally intensive LLM inference is executed offchain, while the result and the commitments or verification artifacts required by the protocol are posted onchain. The chain verifies and settles the result under a deterministic execution specification, with onchain replay available when a computation is disputed.

The central idea is simple: for a fixed model version, fixed weights, fixed tokenizer and runtime, fixed input, and fixed decoding policy, there is a canonical result. In Confirmation mode, an executor produces that result offchain and posts it for onchain verification and settlement. If challenged, the disputed portion can be replayed and adjudicated onchain. In Replay mode, execution itself is performed fully onchain. The current Baranos deployment targets end-to-end result and challenge resolution on the order of approximately one minute, subject to workload and benchmark conditions.

This changes the trust model for AI. Instead of trusting a model host to have run the claimed model correctly, an application can rely on a public execution specification and an optimistic dispute mechanism. The happy path is efficient because inference runs offchain; correctness remains enforceable because every protocol component can run onchain and any disputed component can be replayed there. The settled result can then become blockchain state and be consumed by smart contracts, markets, agents, governance systems, insurance applications, risk engines, and other programs.

01 From Smart Contracts to Verifiable Intelligence

Smart contracts are powerful because their execution is public, deterministic, and shared. The moment an application needs interpretation rather than arithmetic, however, that trust model usually breaks. A protocol calls a private AI API, relies on a committee, asks token holders to vote, or accepts an offchain answer from a privileged operator.

Baranos AI is designed to close that gap. Its objective is to make model inference behave more like a smart-contract state transition: the model is identified, the inputs are committed, the execution policy is fixed, and the output can be verified.

This is particularly important when AI output can move money, resolve a market, grant permission, classify collateral, trigger insurance settlement, or direct an autonomous agent. In those settings, “the API said so” is a weak foundation. Baranos replaces that with a procedure that can be inspected and challenged.

BARANOS THESIS #1

AI becomes materially more useful to blockchains when the execution of intelligence can inherit blockchain-style guarantees.

02 What Deterministic LLM Inference Means

A deterministic inference job has one canonical computational outcome. In practical terms:

> same committed model + same weights/state + same input + same execution rules → same result

THE CORE PRODUCT

Baranos AI is not best understood as "a chatbot on a blockchain." It is a trust-minimized execution layer for AI inference. Its normal Confirmation mode executes inference offchain and verifies/settles the result onchain using an optimistic challenge mechanism; disputed computation can be replayed onchain, while Replay mode provides fully onchain execution.

This is stronger than receiving a similar answer twice. For a blockchain-grade system, determinism means that independent parties can agree on the exact computation that should have occurred.

2.1 Why ordinary LLM inference can vary

- Sampling can deliberately introduce randomness through temperature, top-p, top-k, or other decoding choices.
- Different numerical formats, kernels, hardware paths, parallel schedules, or runtime implementations can produce small arithmetic differences that compound through generation.
- A provider can silently change weights, tokenizers, system prompts, quantization, context construction, or decoding defaults.

2.2 How Baranos defines a canonical job

A Baranos job binds the variables that determine inference: model commitment, tokenizer and runtime version, numerical/quantization profile, prompt construction, input/evidence commitment, decoding parameters, seed where relevant, context/output limits, and execution policy. Oracle-grade policies can require greedy or otherwise deterministic decoding.

The result is no longer an informal request to "ask an AI." It is a defined computation over committed state.

03 Verification, Challenges, and Fast Resolution

The defining architectural insight is to separate execution from settlement without giving up enforceability. Baranos uses an optimistic design rather than a zero-knowledge proof system: in the normal Confirmation path, expensive inference executes offchain and its result is posted and verified onchain. The protocol does not require every validator to repeat every multiply on the happy path. Instead, any disputed portion can be replayed onchain under the same committed deterministic specification, allowing the chain to adjudicate the dispute.

3.1 The basic flow

1. An application creates an inference job that commits to the model, inputs, policy, payer, deadline, and execution mode: Confirmation or Replay.
2. In Confirmation mode, an eligible executor performs the inference offchain and posts the result plus the required execution receipt, commitment, or verification artifact onchain. In Replay mode, the computation is executed fully onchain.
3. For Confirmation-mode jobs, the posted result enters an optimistic settlement path: absent a valid challenge it can be confirmed according to the job policy; if challenged, the disputed computation is escalated to onchain replay.
4. The chain adjudicates a challenge by replaying the disputed portion under the committed execution specification and determining which result conforms to that specification.
5. Once confirmed or resolved, the canonical result becomes onchain state that downstream programs can consume.

The current Baranos deployment is designed around approximately minute-scale result/challenge resolution. That latency matters: verification becomes useful for interactive markets and applications rather than only for slow, post-hoc auditing.

> **WHAT VERIFICATION PROVES**
>
> Verification proves execution integrity: “this is the result produced by model M on input X under policy P.” It does not prove that the model’s statement is factually true, unbiased, safe, or wise. A perfectly verified model can still be wrong.

04 Current Deployment: Fogo

Baranos AI is deployed on Fogo. The project selected Fogo because the workload demands a combination of high throughput, low latency, predictable execution, and low transaction cost that is difficult to achieve on conventional blockchain infrastructure.

Fogo is an SVM Layer 1 built around a Firedancer-based validator implementation and performance-oriented consensus. Baranos uses Fogo as its underlying consensus and settlement layer for verifiable inference.

Baranos' current engineering position is stronger: for the present implementation and workload, the team considers Fogo the only blockchain environment tested that simultaneously provides the throughput headroom, economics, and latency needed to return and settle a verifiable LLM result in approximately one minute. That "only blockchain" statement should be read as a project-supplied deployment conclusion, not as an independently established universal benchmark across every chain.

4.1 Why performance matters for AI

- Inference jobs generate more data movement and computational coordination than ordinary token transfers.
- Optimistic challenge systems add commitments, receipts, verification steps, and—only when a result is disputed—onchain replay and adjudication.
- Interactive applications cannot tolerate settlement that takes hours.
- If verification costs are too high, applications fall back to trusted offchain servers and lose the core benefit.

05 Architecture

Baranos separates inference into independently committed objects so that model selection, evidence, policy, execution, verification, and settlement cannot be silently bundled behind an opaque endpoint. A key design property is that every component can execute onchain, even when the normal path runs computationally intensive work offchain.

5.1 Execution modes: Confirmation and Replay

Baranos exposes two execution modes. Confirmation is the normal, optimistic verification mode: inference executes offchain, the result and required proof/receipt artifacts are posted onchain, and the chain verifies and settles the result. If the result is challenged, the disputed portion is replayed and adjudicated onchain. Replay is the heavyweight mode: the relevant computation is executed fully onchain from the outset. This dual-mode design is central to the trust model.

CONFIRMATION MODE

Offchain execution → onchain verification and settlement.
If challenged: disputed portion replayed and adjudicated onchain.

REPLAY MODE

Relevant computation executed fully onchain from the outset.

OBJECT	ROLE
Model Registry	Model identifier, weight commitment, tokenizer/runtime metadata, quantization profile, activation state.
Evidence / Input Registry	Documents, structured inputs, canonical hashes, timestamps, provenance and availability policy.
Inference Policy	Prompt template, decoding rules, context/output bounds, deterministic settings, challenge and settlement rules.
Inference Job	Binds model, input root, policy, query, payer, deadline and execution mode (Confirmation or Replay).
Result Account	Output commitment/plaintext, execution receipt or verification artifact, confirmation/challenge state, replay status and settlement metadata.

Offchain execution improves cost and latency on the happy path, while universal onchain executability provides a credible fallback that keeps executors honest. Conceptually, this resembles an offchain matching engine whose results are settled by a chain, except that Baranos can deterministically replay disputed AI computation onchain. The design is optimistic rather than ZK-based: correctness is enforced through commitments, challenge rights, and replay, not by requiring a zero-knowledge proof of every inference.

06 The Current Model: An Open-Weight LLM

The current Baranos deployment uses an Open Weight LLM. The nationality or origin of the model is not the core protocol property; the important point is that Baranos can bind a specific model version and execution environment into a verifiable job.

This makes the deployment a proof of a broader abstraction: the infrastructure is not inherently tied to one vendor, one model family, one language, or one geography. A future Baranos registry can support multiple models concurrently, provided each model has an explicit commitment, reproducible runtime semantics, and a policy that defines how deterministic execution is achieved.

Historical jobs remain pinned to the exact model and policy under which they settled. Upgrading the preferred model therefore does not rewrite the meaning of previous results.

07 Baranos as an AI Oracle

The first major application category is an AI oracle for questions that cannot be reduced to a price feed. Instead of asking token holders or a centralized operator to decide what happened, an application can precommit to a model, evidence policy, prompt, and deterministic resolution procedure.

- **Prediction markets:** determine whether a natural-language event condition was satisfied from a frozen evidence set.
- **Insurance:** evaluate submitted evidence against precommitted policy language.
- **Governance:** classify proposals, check policy compliance, or generate structured determinations from public records.
- **Corporate-action and disclosure monitoring:** classify filings, announcements, or event evidence.

BARANOS THESIS #2

The protocol does not eliminate interpretation. It makes the interpreter, evidence, and rules of interpretation explicit before settlement.

08 What Other Builders Can Use Baranos For

AI-powered smart contracts

Contracts can condition execution on a deterministic model result—for example, release funds if a committed model classifies evidence as satisfying a defined condition.

Autonomous onchain agents

Agents can reason over protocol state and public information without depending on a private inference endpoint. Their decision engine can be pinned to a public model and policy.

AI oracles

Natural-language, image, document, or mixed-input questions can become settlement inputs when the inference procedure is fixed and verifiable.

Risk engines

Protocols can classify disclosures, collateral descriptions, incidents, counterparties, governance changes, or anomalous activity.

Model marketplaces

Model publishers, executors, challengers, and application developers can occupy separate roles. Applications consume verified results without requiring the model creator to host every inference.

Auditable autonomous organizations

A treasury, DAO, or machine-run organization can expose not only what it decided, but the model version, input state, and policy that produced the decision.

Developer primitives

Classification, extraction, reranking, semantic routing, moderation, structured generation, embeddings, and other bounded model tasks can be exposed as composable onchain services.

09 The Trust Model

Baranos changes what an application must trust. A conventional AI API asks the caller to trust the provider's infrastructure and configuration. Baranos instead uses an optimistic verification model built around a public execution specification, cryptographic commitments, economic incentives, and onchain challenge/settlement. The normal Confirmation path executes inference offchain, but the chain can replay and adjudicate any disputed portion. Replay mode can execute the computation fully onchain when that stronger, heavier-weight path is desired.

9.1 Fogo Consensus, Validator Governance, and the FOGO Token

Baranos does not ask FOGO holders or Fogo validators to decide what an AI model should answer. The model commitment, inputs, runtime, and deterministic inference policy define the computation. In Confirmation mode, executors compute offchain and post results onchain; challengers can dispute a result, causing the relevant computation to be replayed and adjudicated onchain. Replay mode performs the computation fully onchain. Fogo provides the underlying consensus and settlement layer that determines which protocol state is canonical.

This distinction matters for AI governance. Fogo validators are not intended to vote on individual model outputs or substitute social judgment for deterministic execution. Their role is lower in the stack: they secure the chain on which Baranos jobs, results, challenges, slashing events, model-registry changes, and protocol upgrades are recorded.

SYSTEM	WHAT THE APP TRUSTS	WHAT CAN BE AUDITED
Private AI API	Provider + hidden configuration	Usually limited
Human / token oracle	Committee or economic voting incentives	Votes and published evidence
Baranos AI	Committed model/policy + optimistic verification + onchain replay fallback	Model, inputs, policy, result, challenge and replay path

Fogo uses proof-of-stake consensus in which validator voting weight is based on FOGO stake, including delegated stake. Fogo also operates with a limited validator set subject to minimum stake and operational requirements. Its published architecture describes validator governance as a mechanism for enforcing network health, including the ability to remove persistently underperforming or harmful validators. Protocol-level governance can require a two-thirds supermajority of staked tokens.

For Baranos, this creates a second security layer around verifiable inference. The Baranos protocol determines whether an inference was executed correctly; Fogo consensus determines the canonical history in which that determination settles. If Baranos becomes infrastructure for autonomous agents, markets, treasuries, or other economically consequential systems, the economic security of Fogo therefore becomes part of the security budget of onchain intelligence.

FOGO holders can participate in that security model by staking or delegating FOGO to validators. This should not be described as a direct vote over what Baranos is allowed to think or say. A more precise formulation is that FOGO provides economic participation in, and security for, the validator infrastructure that ultimately settles Baranos state. In that sense, Fogo supplies consensus over what the AI computation actually was—not editorial control over what the AI ought to conclude.

BARANOS THESIS #3

Fogo does not decide what the AI thinks. It provides consensus over what the AI actually computed.

10 Security and Failure Domains

Verifiable execution does not remove model risk. Baranos treats the following as separate security domains:

Model integrity	weights and runtime artifacts must match their registered commitments.
Prompt and policy integrity	instructions, decoding settings, and challenge rules must be versioned and immutable for settled jobs.
Input integrity	every document or data item used by the model must be committed under the job's input root.
Execution integrity	the accepted output must conform to the canonical computation.
Data availability	applications must define how long raw evidence remains retrievable.
Prompt injection and adversarial evidence	untrusted content must be treated as data rather than privileged instructions.
Model risk	hallucination, bias, weak reasoning, adversarial examples, and domain error remain possible even when execution is perfect.
Resource safety	job bounds are needed to prevent adversarial contexts from monopolizing compute or challenge capacity.

11 Economics

A verifiable AI network must make the honest path economically preferable to both trusted centralization and adversarial execution. Jobs therefore need explicit pricing for execution, storage, verification, challenge capacity, and optional data availability.

The developer abstraction should remain simple: specify the model job, latency/security policy, and maximum fee; the protocol handles the eligible execution and settlement path. Fogo's low-latency, high-throughput design is central to the current Baranos implementation because excessive chain fees or congestion would erase the advantage of trust-minimized inference.

11.1 FOGO as the Settlement and Security Asset

Baranos inference creates native demand for Fogo blockspace. A typical Confirmation-mode optimistic flow can require an onchain job request, a posted completion and verification artifact, and—only when necessary—a challenge followed by onchain replay of the disputed computation and dispute resolution. Replay-mode jobs are heavier because execution is fully onchain. Downstream applications may then execute additional transactions based on the settled result.

Those transactions consume FOGO-denominated gas. Under Fogo's published fee design, base transaction fees are low, while users can attach priority fees when faster inclusion has greater economic value. Priority fees accrue to the block producer. This is especially relevant to AI workloads because the value of latency is heterogeneous: a low-value classification job may tolerate delay, while a liquidation, arbitrage, market-resolution, or autonomous-agent decision may rationally pay more for urgent inclusion.

Baranos can therefore create two distinct economic markets: an inference market that compensates executors and challengers for correct computation, and a blockspace market in which FOGO pays for settlement, security, and latency. These layers should remain conceptually separate. A future Baranos-specific asset, stablecoin bounty, or other payment mechanism could price AI computation without replacing FOGO's role as the native gas and staking asset of the underlying chain.

If Baranos usage scales, the relevant FOGO demand is not simply the number of AI users. It is the aggregate chain activity generated by requests, completions, challenges, registry operations, and AI-directed downstream actions, multiplied by the mix of base and priority fees those applications are willing to pay. This relationship should be measured rather than assumed and included in future public benchmarks.

12 Performance and Benchmarking

The central Baranos claims are measurable and should be published as reproducible benchmarks. In particular, the project should disclose enough information for third parties to reproduce Confirmation-mode happy-path jobs, challenged jobs with onchain replay, and full Replay-mode execution.

BENCHMARK	METRIC	REQUIRED DISCLOSURE
Inference latency	Time from accepted job to posted result	model, precision, context, output tokens, executor hardware
Challenge resolution	Time from challenge to canonical settlement	challenge mechanism, disputed unit, verifier hardware, network conditions
Cost	Total chain + execution + verification cost	fee assumptions, compute provider, storage/data availability
Determinism	Bit-/token-level agreement across independent runs	runtime, kernels, numerical format, decoding policy
Concurrency	Jobs/sec and p95/p99 latency under load	network load, model mix, scheduler policy

CURRENT TARGET

Baranos is deployed on Fogo and is designed to return a Confirmation-mode result and, where challenged, resolve it through onchain replay in approximately one minute. Public benchmark artifacts should separately define Confirmation happy-path latency/cost, challenged replay conditions, and full Replay-mode execution, including model size, context, output length, hardware, fees, and finality.

13 Why This Is More Than "Onchain AI"

Running every neural operation onchain on every job is not the breakthrough. The important combination is deterministic specification, optimistic offchain execution on the happy path, onchain verification and settlement, challengeability, onchain replay of disputed computation, an optional fully onchain Replay mode, fast settlement, and composability.

If those properties hold, the output of intelligence becomes something another program can safely depend on. A smart contract can bind itself in advance to a model's canonical answer in much the same way it binds itself to ordinary program logic.

That turns Baranos from a model demo into infrastructure. The long-term opportunity is a permissionless intelligence layer in which models are registered, inference can be executed competitively offchain under Confirmation mode, any disputed computation can be replayed onchain, Replay mode remains available for fully onchain execution, and settled results become shared state.

14 Roadmap

Phase 0 — Fogo deployment	Operate the current Open Weight LLM on Fogo; validate Confirmation-mode offchain execution and onchain verification, challenged onchain replay, full Replay mode, settlement latency, determinism, and cost.
Phase 1 — Public benchmarks	Publish reproducible end-to-end measurements, model/runtime commitments, challenge traces, and cost disclosures.
Phase 2 — Developer primitive	Stabilize Model, Input, Policy, Job, Result, and challenge interfaces; release SDKs for SVM developers.
Phase 3 — AI oracle	Launch reference natural-language resolution applications with frozen evidence and deterministic policies.
Phase 4 — Multi-model registry	Support additional open-weight models with explicit versioning and historical pinning.
Phase 5 — Permissionless verifiable intelligence	Open executor/challenger participation and allow any Fogo application to consume settled inference as a native dependency.

15 Conclusion

Blockchains turned execution into a public fact. Baranos AI applies the same principle to machine inference.

The model is known. The inputs are known. The execution policy is known. The intended computation is deterministic. In the normal Confirmation mode, inference executes offchain and the result is verified and settled onchain through an optimistic mechanism. If challenged, the disputed portion can be replayed and adjudicated onchain. Replay mode can instead execute the computation fully onchain. A settled output can then be consumed by another program.

Baranos AI is deployed on Fogo because the project requires unusually high throughput, low cost, and low latency. For the current workload, the team's engineering assessment is that Fogo is the only tested blockchain environment capable of satisfying all three requirements while targeting approximately minute-scale result and dispute resolution.

If the deployment continues to validate those properties under reproducible load, the result is not simply an onchain LLM. It is a new blockchain primitive: optimistic, verifiable intelligence that can use efficient offchain execution without asking software to trust the machine or company that produced the answer, because disputed computation remains reproducible onchain.

The deployment also creates an important separation of responsibilities. Baranos governs the specification and verification of deterministic intelligence; Fogo supplies canonical settlement and proof-of-stake economic security. FOGO holders and delegators therefore do not choose individual AI answers, but they participate economically in the validator layer that secures the history in which those answers become authoritative onchain state. If autonomous systems increasingly depend on Baranos, that settlement layer becomes part of the governance and security architecture of verifiable intelligence.

REFERENCES

Fogo protocol references for Sections 9.1 and 11.1: Fogo Architecture documentation (docs.fogo.io/architecture.html), Fogo Protocol Litepaper v2 (api.fogo.io/litepaper.pdf), Fogo whitepaper (fogo.io/whitepaper.pdf), and FOGO Tokenomics (fogo.io/blog/introducing-fogo-tokenomics--performance-without-compromise).

END STATE

A smart contract should be able to call intelligence with the same confidence that it calls code.