Fogo: A High-Performance SVM Layer 1

Version 2.0

Abstract

This litepaper introduces Fogo, a novel layer 1 blockchain protocol delivering breakthrough performance in throughput, latency, and congestion management. Fogo is an adaptation of the Solana protocol that introduces a globally accessible settlement layer using zoned consensus and standardized high-performance validation to deliver fast confirmations and low fees.

Fogo maintains compatibility with the Solana Virtual Machine (SVM) execution layer, allowing existing Solana programs, tooling, and infrastructure to migrate seamlessly while achieving significantly faster transaction settlement.

1. Introduction

There is tremendous economic value in an ownerless global computer. Ethereum, Solana, and other smart contract blockchains have unlocked a significant amount of activity and applications even while critics have dismissed blockchains as "slow databases." Blockchains have gotten dramatically better at the things they can control: leader selection, vote aggregation, fork choice, runtime efficiency, and recovery under stress. Yet end-to-end performance is increasingly dictated by what protocol designs to date have ignored: network distance and tail latency. Blockchain designs to date have typically abstracted–or ignored, implied, or assumed without mitigating–the fact that the protocol must be deployed on a planet-sized network governed by physics, routing, and machines that are never identical.

Two constraints keep reappearing across every "fast chain" design, whether the designers acknowledge them explicitly or not:

1. Latency is not a nuisance; it's the base layer.
2. Distributed performance is dominated by the slowest tail, not the average node.

These constraints are not inconveniences; they are the environment. Many protocol designs implicitly assume them away then spend enormous effort squeezing marginal gains out of higher consensus and protocol layers. Fogo takes the opposite approach,

confronting the physical constraints directly. That shift unlocks practical, significant improvements in finality and throughput.

The Speed Limit of the Internet

Signals propagate through fiber at roughly 200,000 km/s (≈2/3 the speed of light in vacuum), implying a ~100 ms one-way transit time to go halfway around Earth's circumference in an ideal straight path. Real networks are not straight lines on a globe; they follow submarine cable topology, peering economics, congestion, and packet processing overhead. In practice, round-trip times (RTTs) depend on routing and congestion; transatlantic RTTs are often on the order of ~70–90 ms, and New York–Tokyo RTTs are often ~170 ms.

Meanwhile, most consensus protocols require multiple messages across a quorum to commit a block. The classic Byzantine fault tolerant consensus pipeline has distinct phases (e.g., pre-prepare/prepare/commit) precisely because agreement is achieved by exchanging authenticated votes among replicas. With a dense and globally distributed network, the overwhelming majority of block settlement time is consumed by internode communication. Hence, the performance of even the most efficient consensus algorithm is unavoidably dependent on network delay.

Operation	Approx. Latency in ns ($1x10^{-9}$)
L1 memory cache reference	1
Main memory reference	100
Read 1MB sequentially from memory	10,000
Round trip within same datacenter	100,000
TCP packet round trip between continents	150,000,000

Because messages propagate with delay, different parts of the network learn about new state updates at different times. Temporary disagreement is therefore not an edge case; it is the default condition that any high-throughput chain must continuously resolve. Blocks bundle coordination into discrete updates, and consensus protocols are, at bottom,

structured vote exchanges that converge the network back to a single history. The faster you try to go, the more you are fighting propagation delay and variance in who has seen what. Before a block can be "final" socially, it has to be known physically.

Where blockchain designs are oblivious to network distance, they must necessarily accept slower updates. On the other hand, if you start from a strong consensus protocol and merely reduce the physical distance data needs to travel before it can be finalized on the chain, you can meaningfully improve network latency.

Fogo Thesis #1: All else equal, a blockchain that is aware of physical space can be faster than one that is not.

Client Performance

In large-scale interactive systems, the enemy is not average latency—it's tail latency: the slowest few percent of operations that end up dominating user-perceived performance and throughput at scale. That dynamic gets even sharper in decentralized systems, where an end-to-end operation is not a single request/response, but a distributed workflow spanning many independent machines. In that regime, the critical path is set by the slowest components you must wait for, not the typical one.

In a typical blockchain design, a block is committed only after enough parties have processed it, voted, and propagated votes. This creates a "weakest link" phenomenon: in a quorum-based protocol, the network doesn't depend on every validator being fast, but it does need the quorum threshold to be reachable reliably within the target latency window. If validator performance is subject to wide variance—different client implementations, different tuning, different hardware, different network quality—then the chain's real-time behavior is governed by the distribution of those choices, not the elegance of the consensus design. Where blockchain designs are oblivious to validator performance, they must necessarily accept slower updates.

Fogo Thesis #2: A blockchain that requires high performance validator implementations can be faster than one that is not.

2. The Fogo Blockchain

Fogo takes these physical constraints seriously and confronts them head-on. It builds on the protocol and execution model pioneered by Solana, targeting maximal compatibility with the Solana Virtual Machine (SVM). Where Fogo differs is in what it treats as first-class design parameters: (1) the geographic and network topology that messages must traverse, and (2) the real distribution of validator performance in the wild.

Concretely, Fogo pursues low latency and high throughput through two choices that follow directly from the constraints above:

- **Localized consensus:** reduce the distance and dispersion of the quorum on the critical path, dramatically reducing delays from wide-area network latency.
- **Performance enforcement:** reduce variance by standardizing on a highly optimized validator implementation and explicit operational requirements, so the network's behavior is governed less by the slowest outliers and more by a predictable quorum path.

The rest of this paper describes the Fogo architecture, including validator zones and epoch activation, the Firedancer-based validator implementation, and Fogo's fee model, inflation, built-in programs, and Sessions standard.

3. Architecture

Fogo implements the Solana Virtual Machine (SVM) through the open-sourced Firedancer validator client software and has been designed to be maximally backwards compatible with Solana, including with block propagation, SVM execution, and the other major components of the Solana protocol. This allows Fogo developers to easily integrate and deploy existing programs, tooling, and infrastructure from the Solana ecosystem, as well as benefit from continuous upstream improvements in Solana.

Like Solana, blocks are proposed by a rotating leader validator that is selected through a deterministic, stake-weighted algorithm. The leader schedule is computed at epoch boundaries from the ledger state, with slot leadership assigned according to a stake-weighted schedule (validators with more stake are scheduled for more slots), using a PoH-seeded, stake-weighted ordering computed in advance of the epoch. This schedule is deterministically computed using a protocol-defined pseudorandom seed derived from chain state, ensuring that the rotation is predictable yet fairly distributed across the validator set.

During their designated slot, the leader validator receives transactions through a QUIC-based ingress pipeline, validates signatures, executes transactions to update state, and packages the results into entries that are cryptographically linked to Proof of History (PoH). These entries are then fragmented into shreds and broadcast to the network through Turbine, Solana's block propagation protocol, which organizes validators into a tree structure for efficient distribution.

Consensus is achieved through Tower BFT, a Byzantine fault tolerant algorithm that uses exponentially increasing lockout periods on validator votes. As validators vote on forks, they

commit to those chains with lockouts that double with each successive vote, creating an economic cost to switching forks that grows super-linearly. The network selects the canonical chain using a heaviest fork choice rule, where the accumulated stake weight of validator votes determines which fork to build upon. A block is considered confirmed once 66%+ of stake has voted for it on the majority fork; it is considered finalized once it reaches maximum lockout, commonly represented as 31+ confirmed blocks built atop it.

3.1 Validator Zones

Fogo introduces a novel validator zone system that extends Solana's consensus model to enable geographic and temporal partitioning of the validator set. This mechanism allows the network to organize validators into distinct zones, with only one zone actively participating in consensus during each epoch.

Validators are assigned to zones, with each zone containing a subset of the total validator population. Zone definitions and validator assignments are stored on-chain as Program-Derived Accounts (PDAs) managed by a dedicated Zone Program, enabling transparent governance and configuration of the zone system.

During each epoch, the protocol selects a single active zone using a deterministic algorithm. Only validators within the active zone are eligible to propose blocks, vote on forks, and participate in consensus for that epoch. This is enforced through stake filtering at the epoch boundary, where vote accounts and stake delegations for validators outside the active zone are excluded from the epoch's consensus participation set.

Fogo supports multiple zone selection strategies controlled by an on-chain algorithm identifier:

- Epoch-based rotation: Zones rotate sequentially based on the epoch number, ensuring each zone receives proportional time as the active consensus set.
- Follow-the-sun rotation: Zones can activate based on UTC time rather than epoch boundaries, enabling the network to shift consensus activity across geographic regions throughout a 24-hour cycle. Each zone is assigned a duration in milliseconds, and the protocol calculates which zone should be active by determining the elapsed time since a configured daily start point. This approach allows Fogo to maintain active consensus during peak hours in different regions, potentially reducing latency for users in the currently active geographic zone.

When stake filtering occurs at the epoch boundary, only validators in the active zone contribute to:

- The stake-weighted leader schedule for block production
- Tower BFT voting and fork choice calculations
- The total network stake used for computing supermajority thresholds

Validators in inactive zones remain connected to the network and continue to sync blocks, but they do not propose blocks, vote on forks, or earn consensus rewards during epochs when their zone is inactive. This creates a rotating validator model where different subsets of the network alternate responsibility for maintaining consensus.

To ensure network security, zone configurations include a minimum stake threshold parameter that filters out zones with insufficient total delegated stake. Only zones that meet the threshold can become active, preventing consensus from being controlled by a zone with too few or too lightly staked validators. The protocol calculates total stake for each zone by summing stake delegations across all validators in that zone, ensuring that active zones maintain adequate security properties.

This zone-based architecture enables Fogo to establish a network topology that accounts for physical space while maintaining compatibility with Solana's core consensus protocol within each active zone.

3.2 Validator Client

Firedancer is a next-generation validator client engineered by Jump Crypto for high-performance consensus. The design includes handling for tail latency, bursty demand, and adversarial operating environments. The Fogo mainnet validator implementation is an intermediary production client known as "Frankendancer," a hybrid client where Firedancer components (notably networking and block production while leader) run alongside Agave code.

The Frankendancer validator is decomposed into independent functional units called "tiles," each running as a separate sandboxed process pinned to a dedicated CPU core. Rather than sharing CPU resources through context switching, each tile is designed to maximize CPU utilization by continuously processing its specific workload in a tight loop. This design reduces scheduler-induced jitter by pinning tiles and using tight polling loops, improving predictability under load.

The core transaction processing pipeline consists of these primary tiles:

- Net tile: Receives network packets using AF_XDP (Address Family eXpress Data Path), a Linux kernel feature that enables zero-copy packet I/O directly from network interface cards.

- QUIC tile: Processes the QUIC transport protocol, reassembling transaction streams from network packets and forwarding complete transactions downstream.
- Verify tiles: Perform cryptographic signature validation on transactions. This stage can be parallelized across multiple cores, with each verify tile independently processing a subset of incoming transactions.
- Dedup tile: Eliminates duplicate transactions using hashes produced during signature verification, ensuring each unique transaction is processed only once.
- Resolv tile: Provides address resolution and integration with Agave's validator logic for transaction validation.
- Pack tile: Aggregates validated transactions into microblocks when the validator is scheduled as leader, optimizing block packing for maximum fee revenue and efficient execution.
- Bank tile: Executes transactions against the current account state, updating balances and invoking smart contract programs.
- PoH tile: Maintains the Proof of History chain, providing the cryptographic clock that timestamps all validator operations.
- Shred tile: Encodes completed blocks into shreds using Reed-Solomon forward error correction, enabling efficient distribution and recovery.
- Store tile: Persists shreds to the ledger database for long-term storage and replay capability.

Tiles communicate through shared memory message queues implemented by Firedancer's Tango system. Rather than copying data between tiles, transactions and blocks remain in fixed memory locations while tiles pass lightweight metadata pointers. This zero-copy approach eliminates memory bandwidth bottlenecks and reduces latency, as data traverses the entire pipeline without serialization or deserialization overhead.

This architecture achieves high throughput through several mechanisms:

- Parallelism: Independent operations (particularly signature verification) scale linearly across multiple cores. A validator can dedicate four or more cores to signature verification, processing transactions in parallel rather than sequentially.
- Predictable execution: Pinning each tile to a dedicated core eliminates context switch overhead and cache pollution. Each core's instruction and data caches remain hot for that tile's specific workload.
- Zero-copy data flow: Transactions and blocks pass through the pipeline in shared memory without copying, reducing memory bandwidth consumption and latency.
- Kernel bypass: AF_XDP can bypass large parts of the kernel networking stack; with driver and configuration support, it can enable a zero-copy fast path and materially reduce per-packet overhead.

- Cache-friendly design: Each tile's tight execution loop fits in the CPU's instruction cache, and frequently accessed data structures fit in L1/L2 data caches.

The tile architecture enables Firedancer to approach the theoretical limits of hardware performance by eliminating software inefficiencies that plague traditional validator implementations.

3.3 Network Fees and Inflation

Fogo transaction fees are designed to mirror Solana's. A simple transaction with one signature costs 5,000 lamports. During congestion, users typically add an optional prioritization fee (a 'tip') to increase inclusion probability. Half of the base fee is burned and half is paid to the validator that processes the transaction (and may be shared with delegators indirectly via that validator's staking economics). 100% of priority fees go to the block producer.

Like Solana, Fogo also implements a rent mechanism to maintain network sustainability by charging accounts for the storage space they consume. Rent is collected at a rate of 3,480 lamports per byte of storage per year, with half burned and half distributed to validators. This creates an economic incentive for efficient resource usage while preventing state bloat. However, accounts that maintain a minimum balance are rent-exempt and never pay rent. The rent-exempt minimum is typically computed as (data length + overhead) × (lamports per byte-year) × (exemption years), with commonly used parameters being 3,480 lamports/byte-year and a two-year exemption window.

Most users are expected to interact with rent-exempt accounts exclusively and experience rent as a one-time minimum balance requirement rather than an ongoing fee.

Fogo mainnet operates with a fixed annual inflation rate of 2%. This rate represents the terminal inflation target that Fogo's emission schedule was designed to reach and maintain indefinitely. All newly minted tokens from inflation are distributed to validators and their delegated stakers. This ensures that network security incentives remain aligned with active participants who secure the chain.

Rewards are calculated at each epoch boundary and distributed proportionally based on a points system:

- Points calculation: Each stake account earns points equal to its delegated stake amount multiplied by the vote credits earned by its validator during the epoch
- Validator commission: Validators specify a commission rate that determines their share of rewards before distribution to delegators

- Staker rewards: The remaining rewards after commission are distributed to stake accounts proportionally to their accumulated points

This mechanism ensures that validators who actively participate in consensus (earning vote credits by correctly validating blocks and voting on forks) generate higher returns for their delegators, creating economic incentives for both uptime and correct validation behavior.

3.4 Built-in Programs

Fogo's native programs mirror Solana's core set (System/Vote/Stake and loaders). The built-in programs form the foundation layer that enables all higher-level functionality, from basic transfers to complex DeFi applications running as user-deployed programs. Fogo also ships with a token program based on Solana's SPL Token but modified to accommodate Fogo Sessions, as described below.

3.5 Sessions: Enabling Seamless On-Chain Experiences

Fogo Sessions is an open-source standard that enables Web3 applications to deliver user experiences comparable to traditional Web2 applications. Sessions addresses three critical friction points in blockchain applications: wallet compatibility, transaction costs, and signature fatigue.

At its core, Sessions allows users to grant time-limited, scoped permissions to applications through a single signature. This session-based model fundamentally transforms how users interact with Fogo applications, enabling a gasless experience while maintaining self-custody and strong security guarantees. The Sessions mechanism operates through an interaction between on-chain programs and cryptographic intents:

- Session Creation - A user creates a session key by signing a structured message (intent) specifying session parameters including authorized programs, token spending limits, and expiration time. This intent follows a structured, signable authorization message format consistent with other "intents" designs emerging in the industry. The session key is only stored in the browser and is marked as non-exportable to reduce the risk of key extraction under normal browser operation.
- On-Chain Registration - The application submits this signed intent to the Session Manager program, which validates the intent and creates an on-chain Session account. This account stores the authorization parameters, linking the user's primary wallet to the temporary session key.
- Session Usage - With an active session, the user can execute transactions through the temporary session key saved in the browser. The on-chain Sessions program

validates each transaction against the session's constraints (authorized programs, token limits, expiration).

Fogo Sessions also includes optional fee sponsorship functionality whereby application services or third parties can sponsor transaction fees for users. Fee sponsors can use configurable constraint systems to determine which transactions qualify for sponsorship, protecting app developers from abuse. Application developers have a choice as to how they charge their users to pay for transaction fees, whether it is in native tokens, stablecoins, or another token. Sessions provides tools to enable developers to easily implement any of these options.

Fogo's session infrastructure integrates seamlessly with Fogo's built-in SPL Token program through targeted, non-invasive modifications. Rather than redesigning the core token architecture, the enhanced SPL Token program layers session-based authorization onto existing SVM delegation mechanisms, preserving backward compatibility while enabling sophisticated temporary authorization patterns. The modified SPL Token program enables a session key to conduct transfers on behalf of the underlying wallet within the scope of the authority delegated to the session key.

Fogo Sessions has the potential to unlock new application and development paradigms on Fogo, including with trading/DeFi, gaming, mobile, and cross-chain flows.

4. Conclusion

Blockchains have matured from fragile experiments into resilient systems, and the application layer has proven the demand for an ownerless global computer. But modern usage is increasingly sensitive to the constraints of network latency and validator variance. The Fogo thesis is that armchair blockchain consensus designs are at their asymptote, while faster block settlement times are available by optimizing the physical stack. Reducing the distance light must travel and reducing the variance in validator performance can significantly reduce latency and unlock substantial economic value. Fogo's claim is straightforward: a better global computer is reachable by broadening the design space to address the real-world systems and conditions under which blockchains must operate.