TELESCOPE

PROJECT REPORT

Table Of Content

What is Telescope?

Decentralized video streaming

Telescope sits between a video player (like one built with DASH.js) and a network of IPFS nodes. It intercepts requests for video manifests and segments, intelligently processes them, and delivers content from IPFS to the player in a way that mimics how traditional CDNs work—but in a decentralized fashion.

Why?

IPFS segments come from varied nodes, throughput is hard to predict—Telescope's IPFS-aware ABR solves this, improving QoE and reducing stalls.

Vanilla Telescope

> How does it work?

Telescope solves the issue by measuring fetch latency and bandwidth from each IPFS node. It **rewrites** the DASH manifest (MPD) to reflect these estimates, guiding the player to choose bitrates it can actually support—avoiding stalls from slow or overloaded nodes.

- Telescope Estimates Bandwidth (Tc - Tg - Tn)
- Telescope Rewrites the MPD (Manifest)
- Telescope measures fetch times, updates, and serves segments from cache or IPFS
- Telescope “learns” over time as more segments are fetched

Ready for new Telescope?

Methodology & Approach

Re-Design

Original Telescope system was a functional prototype, but not production-grade:

- Clean separation of MPD logic, segment routing, and metrics.
- Bandwidth-aware logic (Tc, Tg, Tn).
- Support for scalable caching and multi-node IPFS.

New Impl

Implementing a clean, modular, high-performance and monitor-able Telesocpe:

- Code modularity → Allowing debugging and future extensions
- Strong Monitoring System → Keep track of performance!
- Identifying bottlenecks -> Using tracing tools such as Jaeger.
- Migrating to Fiber from Gin → Improved HTTP request performance!
- **Stateless Proxy System** → Now HTTP headers carry client metadata!

Scalablity

Telescope is now **SCALEABLE**! It can have multiple smart stateless proxy gateways leading to faster request processing and improving QoE

- Having multiple proxies, requires a new smart ABR algorithm
- **New Statistic-based smart ABR algorithm** is born to answer scalablity
- The new algo uses a weighted formula to consider all conditions
- Horizontal scaling with adding multiple proxy nodes

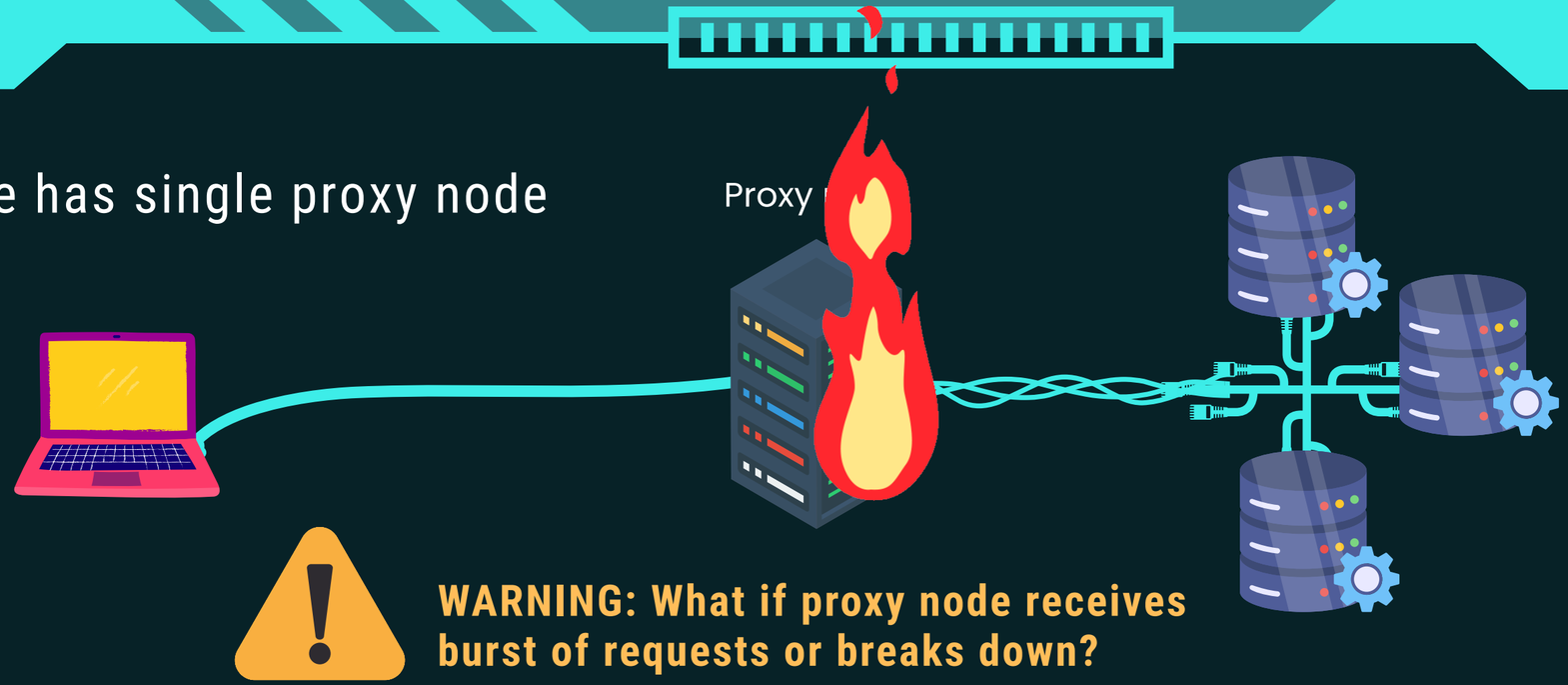
Layered Cache

We introduced a 2-layer caching mechanism to use in-memory and persistent in conjunction (First memory, then disk)

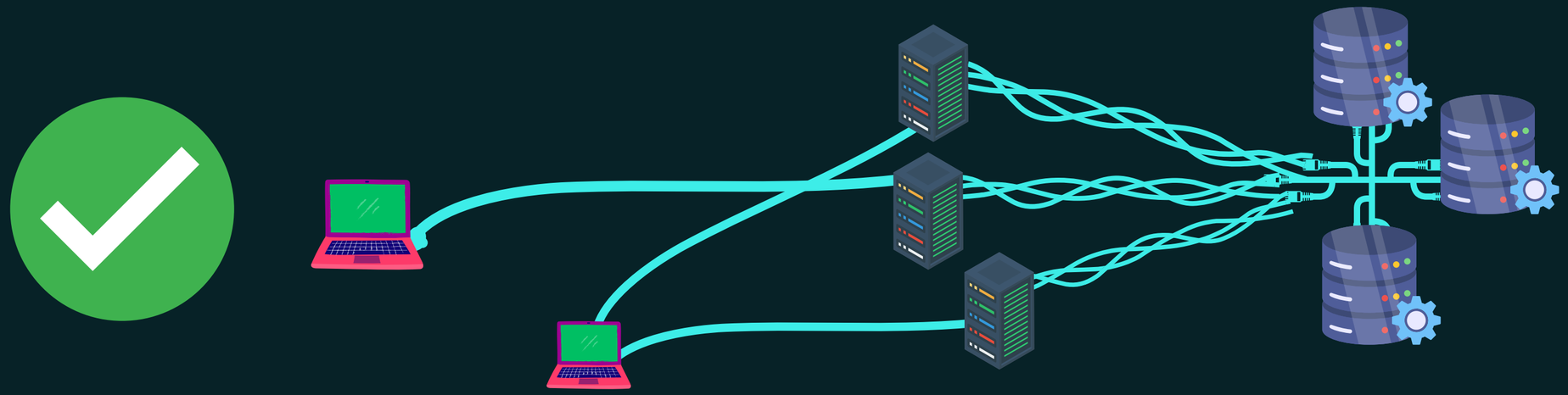
- Developing and adding new **File-based** persistent storage
- Allows segment reuse across different playback session
- Allows flexibility to switch to other storage systems (e.g. S3Stream)
- Rapid repeated access → In-memory cache for recently fetched segments

Let’s Learn Together

Vanilla Telescope has single proxy node



WARNING: What if proxy node receives burst of requests or breaks down?



Let’s Learn Together

Telescope rewrites the manifest based on a gift/punishment system:

Vanilla Telescope ABR Algo

Bandwidth = Tc – Tg (if cached)
Bandwidth = Tc – Tn (if not cached)

WARNING: What if we had several proxies?

New Weighted ABR Algo

Bandwidth = (cached) × 1/3 + (un-cached) × 2/3 (if cached)

Bandwidth = (un-cached) × 1/3 + (cached) × 2/3 (if not cached)

Why Rewriting MPD?

Hey John! How much does it take to jump on a train to get to the Manhattan from Stony Brook? We wanna order foods!

Role: Client (e.g Dash)

Around 2 hours! Let me check the Map application!

Role: Telescope, Re-writing MPD

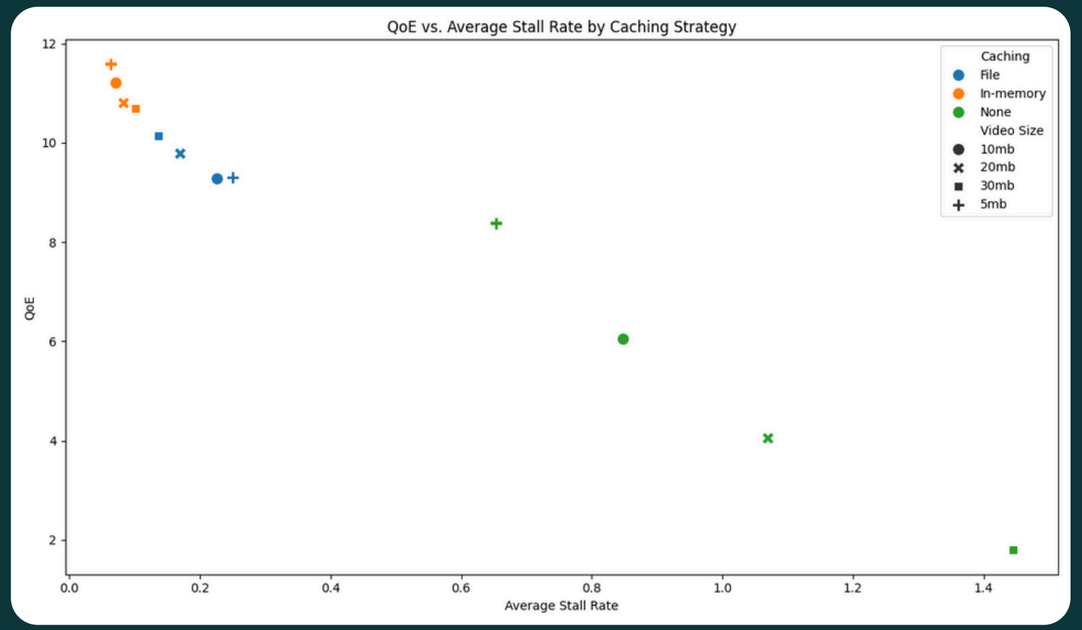
What? 2 hrs? YOU WISHED! My **records** tell me the Stony Brook LIRR is broken and you have to use Ronkonkoma → It takes around 3 hours → **Lower your expectancy**!

WOW, OK, I didn't know that!

Vanilla VS. New Telescope

Feature	Old Telescope	New Telescope
Architecture	Tightly coupled logic	Clean Modular Layers
Web framework	Gin	Fiber (Fast http)
MPD rewriting	Static	Dynamic based on real-time stat
Caching	In-memory With no Policy	2-Layer In-Mem + File based
Proxy Type	State-full	State-less
Scalability	❌	✅
ABR algo	Throughput-based	Smart Statistic-based
Storage Flexibility	Local-only	Swappable
Setup	Simulation	Realistic with clustering
Monitoring System	❌	✅

Finding & Results



Cache Efficiency

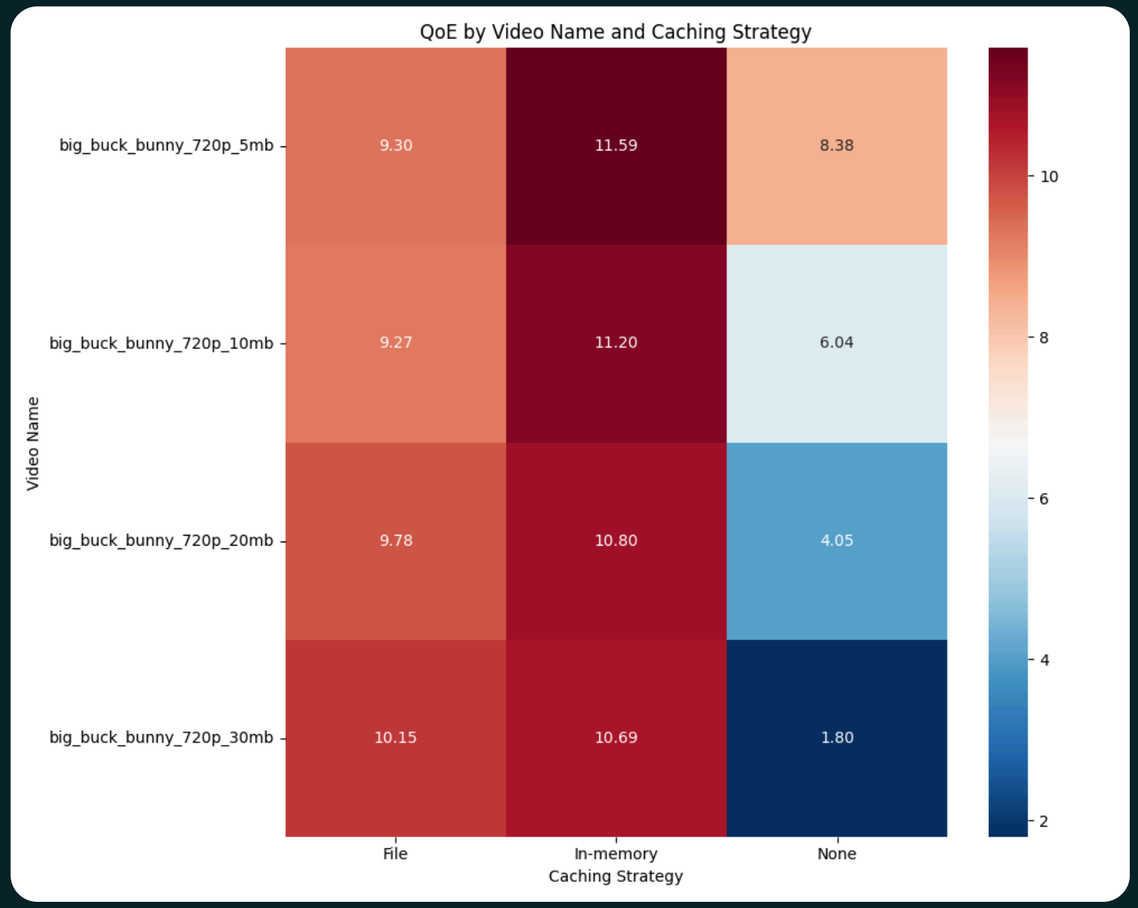
60% HITS

The cache hit rate improved by up to 60% in scenarios with layered caching, leading to AVG of **66% Stall reduction**

60%

66%

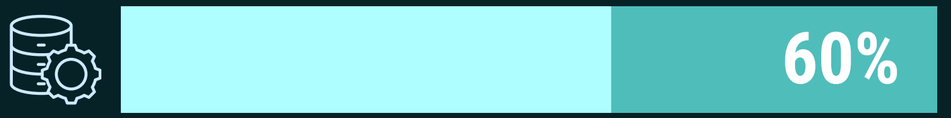
Finding & Results Cont.

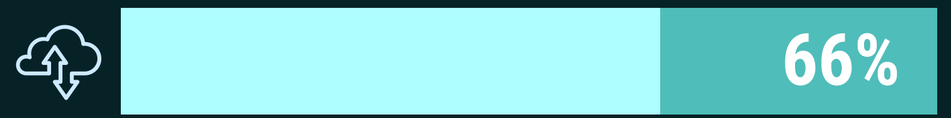


Cache Efficiency

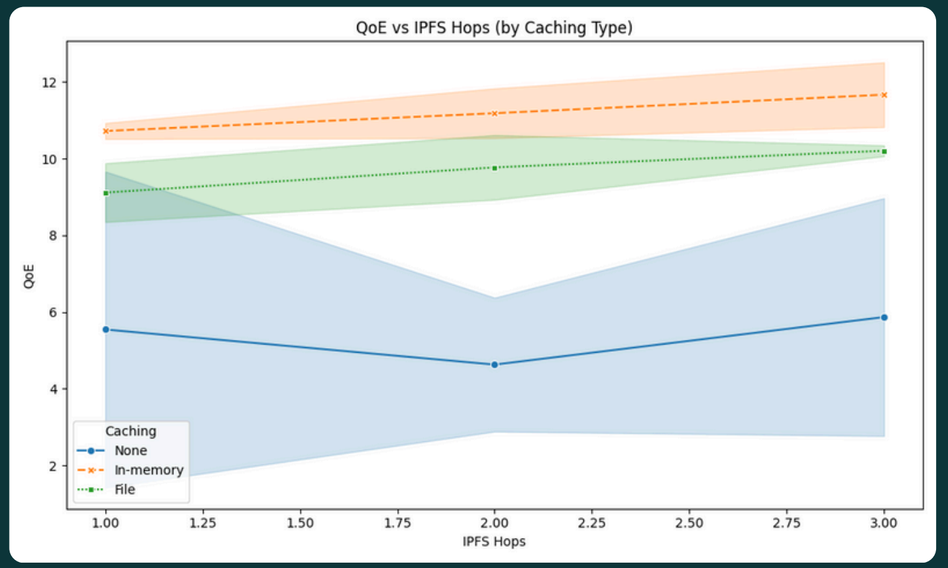
60% HITS

The cache hit rate improved by up to 60% in scenarios with layered caching, leading to AVG of **66% Stall reduction**





Finding & Results Cont.



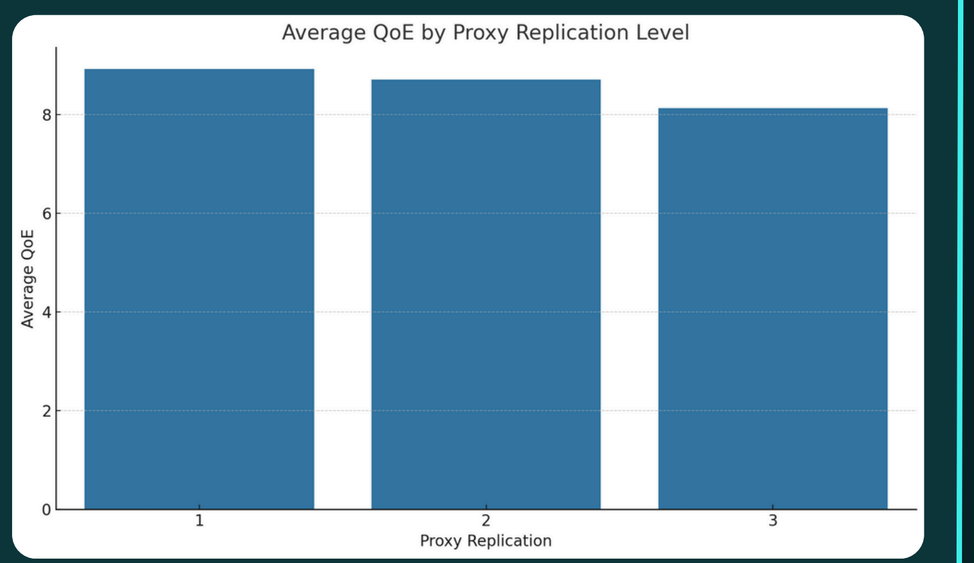
Latency Improvements

45%

Multi-node IPFS retrieval reduced average fetch latency by up to 45%, especially under congestion and high-hop scenarios

45%

Finding & Results Cont.

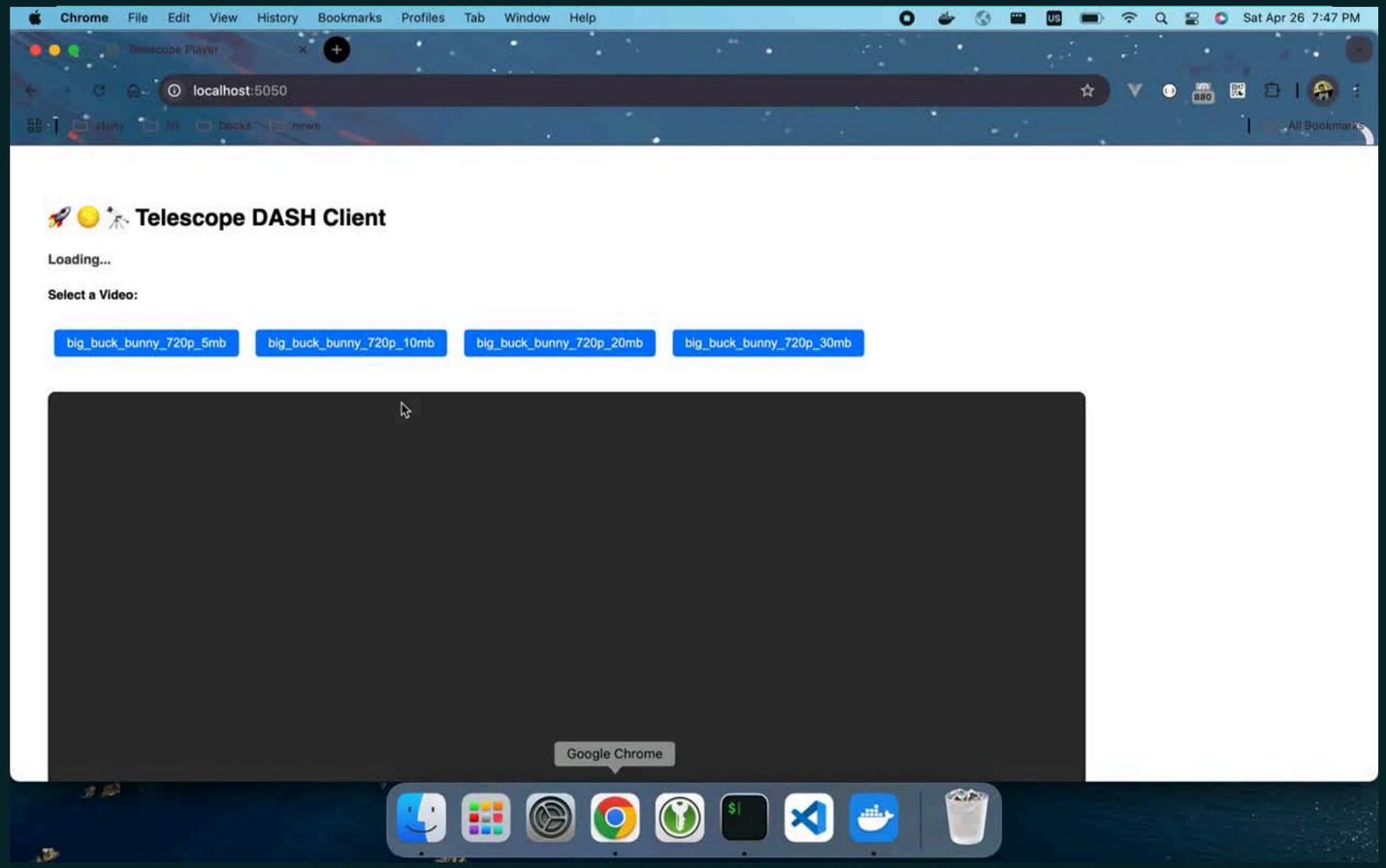


Latency Improvements

45%

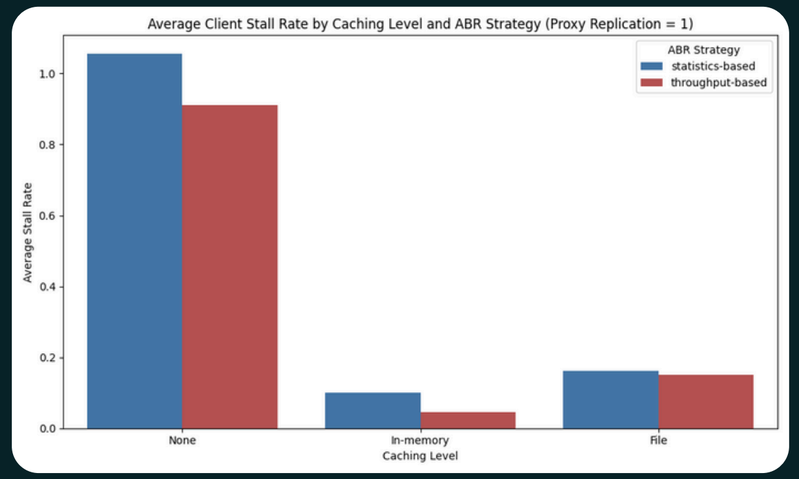
Multi-node IPFS retrieval reduced average fetch latency by up to 45%, especially under congestion and high-hop scenarios

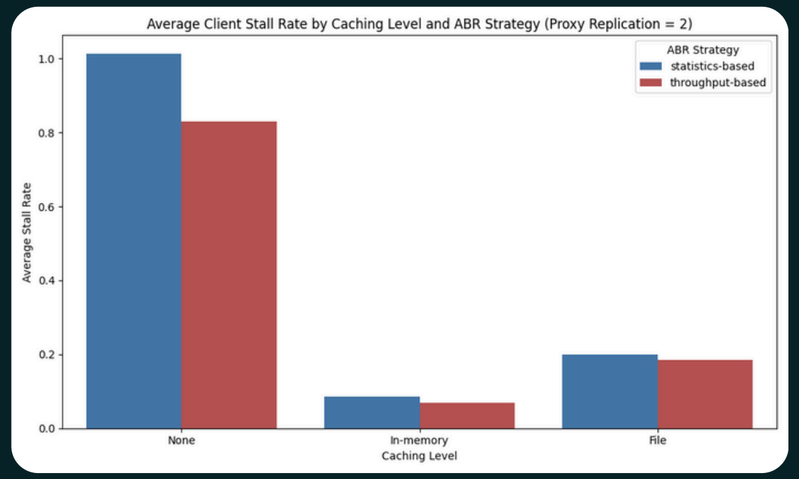
Project Demo

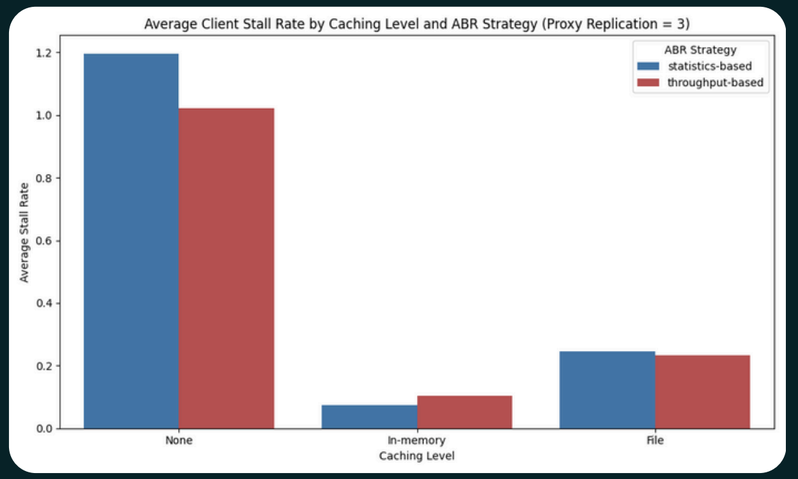


New Telescope

Vanilla Telescope



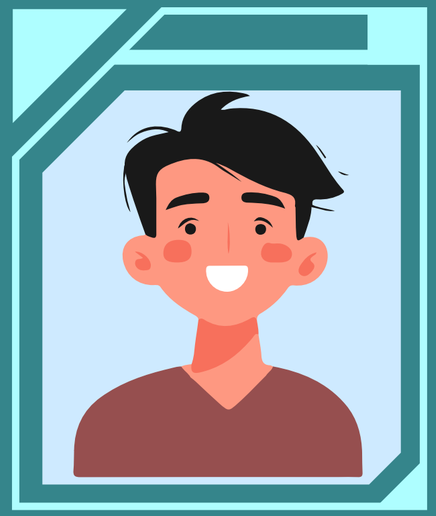




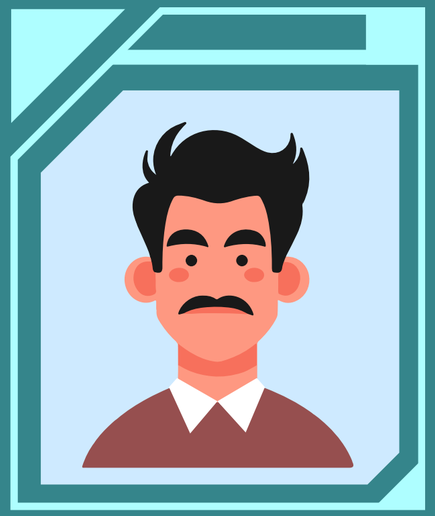
Vanilla VS New Telescope

Average Stall Rate Based on Caching and ABR Strategy with Proxy Replication Levels for 1-3 Replicas

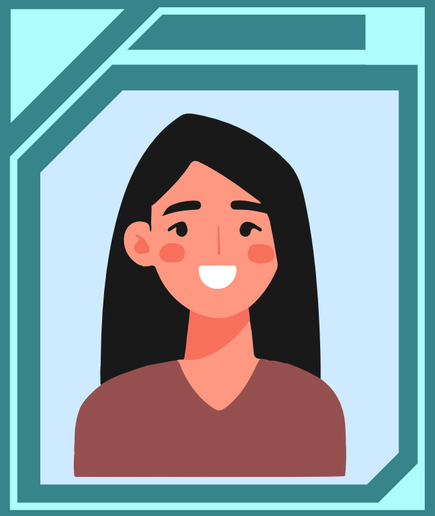
Meet Our Team



Iliya



Amir



Shabnam

THANK YOU

Besides network, we all need a telescope in our life ❤️