Redis + EVM

Supercharging Ethereum calls with in-memory execution

Everton Fraga

Amazon Web Services


aws

PROBLEM STATEMENT

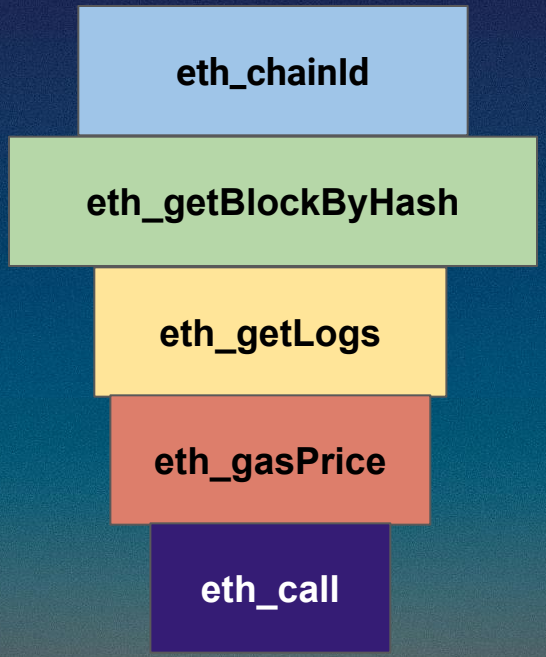
Challenges on RPC scaling

Challenges on RPC scaling

Horizontal scalability of nodes:

- Reaction time — managing snapshots, loading chaindata, sync time
- Forecasting — specialized task. Caveat: over-provisioning
- Final boss: Consistency

Common strategy: caching



eth_chainId

eth_getBlockByHash

eth_getLogs

eth_gasPrice

eth_call

>70%

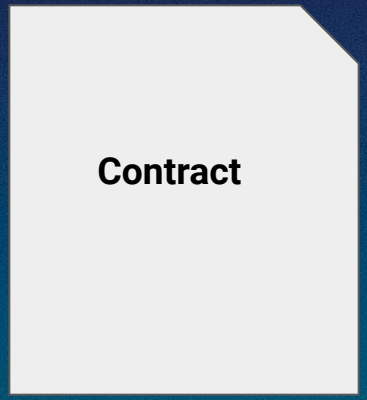
WORD OF THE DAY

Externalization

EVM.lua

- Technical validation ✅
- Micro-EVM interpreter, implemented in Lua
- Executes inside Redis process, minimal storage latency
- Able to process EVM operations (eth_call)

How it works




```
Code: 0x6080604052348015610 01…
Balance: 0x1000000000000000000000000
Nonce: 0x01
Storage:
    0x00…08: 0x67…30f4
    0x00…09: 0x00…a5e4
    0x00…0a: 0x00…e1a9
```

R&D stage

Loads the entire storage from selected contracts. Keeps up with state diff.

Process EVM code strips

EVM compliance

Implement all opcodes, including Transient storage (TLOAD, TSTORE).

Add EVM metering

Deployment

Benchmarks, optimization feedback loop.

Use-case specific features

1.2M rps

Redis on Amazon ElastiCache
(generic benchmark)

>500M rps

In a single
Amazon ElastiCache cluster

EVM.lua

https://github.com/evertonfraga/evm.lua/

Zoom out

Thank you!

Everton Fraga
Amazon Web Services
effraga@amazon.com
@evertonfraga