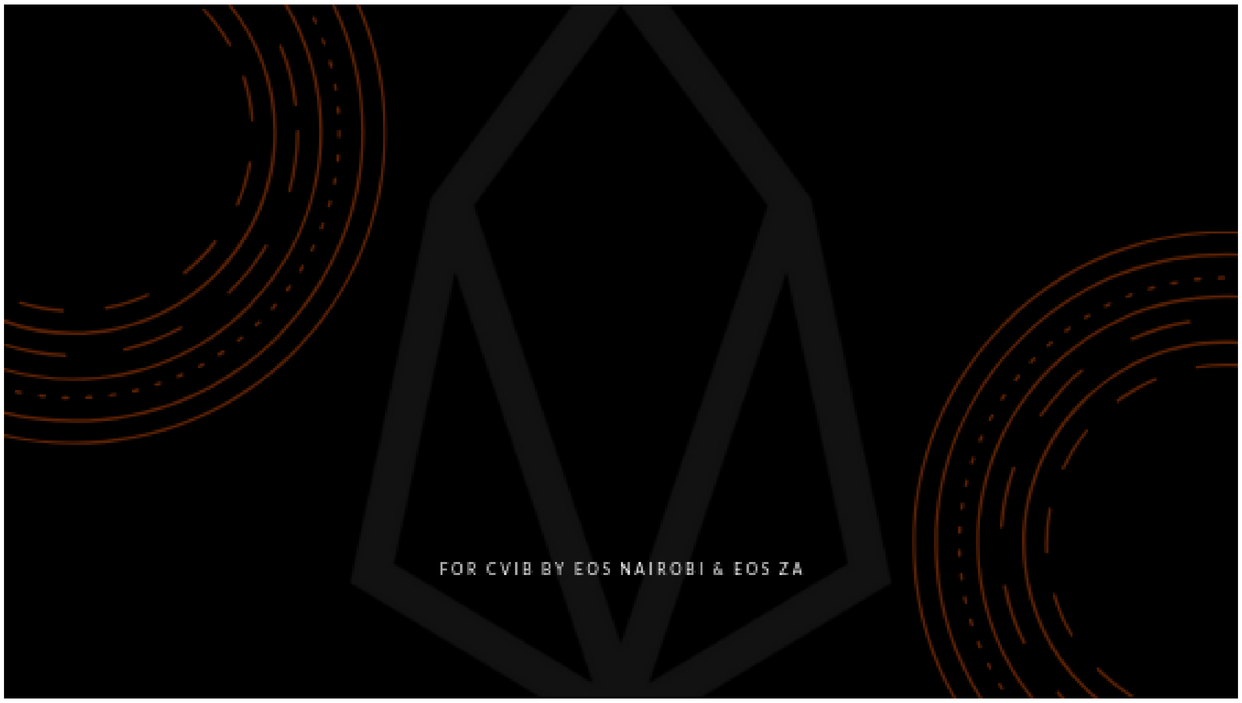

Cassandra Based Full History Solution for EOSIO.

Proposal for the Cooperative Voting Infrastructure Bounties (CVIB)

EOS NAIROBI & EOS ZA PARTNERSHIP

https://t.me/eoszaio

https://t.me/eosnairobi

Background

EOSIO full history services are not critical for the functioning of the EOS blockchain. They are, however, critical for dApps such as block explorers and wallet applications that are essential for mass adoption. The original EOSIO shipped with a proprietary history API served by a nodeos plugin. Although this served its purpose at network launch it has become increasingly difficult to maintain. The original history plugin has now been deprecated. The functions performed by this plugin are now performed by the state history plugin of the MongoDB plugin. Both of these solutions serve the current needs of the EOS ecosystem but suffer from performance and reliability problems. System failures can be recovered by doing a blockchain replay but this is becoming increasingly difficult and time consuming as the blockchain grows.

A number of teams have initiatives to build alternative history APIs. These are likely to serve the needs of many dApps they suffer from two shortcomings;

- In most cases the APIs do not provide the same endpoint as the original history API. The history API forms part of the standard EOSIO documentation and is most likely where most dApp projects will start.
- These initiatives are not open source. This means that the API function will likely become a centralized function. One of the core strengths of blockchain is it’s decentralized nature. To retain its nature all parts of the ecosystem need to have decentralized solutions. Open sourcing these components solved this requirement.

A more scalable and reliable open source solution is required for the general health of the blockchain.

Why Cassandra?

EOS is already the busiest blockchain in the world. Transaction history is growing at a few million transactions per day. At this rate of growth current full history solutions have already reached a point where they are becoming unreliable and difficult to maintain. The only opensource full history solution available today is the MongoDB plugin. If one looks at the history of websites originally built on MongoDB who have experienced massive growth one quickly realizes that MongoDB has scaling difficulties once the database grows beyond a certain point. Most of these projects that have achieved massive scale have transitioned to Cassandra at some point. One can assume that the MongoDB plugin on the EOS blockchain will run into the same issues if EOS adoption reaches the levels we all hope for.

For platforms who have ported to Cassandra the reasons given include;

- Cassandra offers true horizontal scaling. As the cluster grows performance improves.
- Cassandra query performance is consistent and predictable (i.e. the same query will return repeatedly within approximately the same time). This results in a consistent user experience.
- All Cassandra nodes are the same. All that is needed to expand the system is to add new nodes to the cluster. Similarly nodes can be removed by retiring them from the cluster. Data is rebalanced within the cluster as nodes are added and removed.
- The Cassandra architecture includes redundancy by design. The cluster is initialized with a desired redundancy factor. The system itself manages data replication to achieve the desired redundancy levels. If a node failure is experienced the cluster will rebalance to maintain the desired redundancy level.
- Because Cassandra handles redundancy at a system level nodes can be built with commodity hardware which need not include node level redundancy (use RAID, dual power supplies, etc). The financial savings achieved by utilizing commodity hardware and not needing node level redundancy mitigate the cost of storing multiple copies of data in the system.
- Cassandra nodes employ data compression when storing data. This compression results in raw storage requirements being roughly the same as would be required to store uncompressed data without compression for sensible redundancy factors. (i.e. nett result of compression plus redundancy is approximately nill) This is a general statement and is obviously dependent on redundancy factor used and compressibility of raw data.
- Cassandra is rack and data center aware. Nodes can be distributed in multiple racks and in multiple data centers. Data will is automatically distributed efficiently between remote nodes.

Other initiatives

EOS Chronicle Project

This project is a sponsored EOS worker proposal to build a scalable history solution for EOSIO. The proposal considers using ScyllaDB (a re-implementation of Apache Cassandra in C++) for DB data storage.

Project scope

This project scope is to provide a Cassandra based solution for the EOS full history API that conforms to API endpoint defined in the developers documentation for EOSIO (https://developers.eos.io/eosio-nodeos/reference). The history API provides the following end points;

1. get_actions
2. get_transaction
3. get_key_accounts
4. get_controlled_accounts

For each of these endpoints the API should return results that are compatible with the original EOS history API. Completion of this scope will be judged on accuracy of data returned by API endpoints when compared with the original history API.

Evaluation will also include latency and load benchmark tests.

Technical references

A considerable amount of work has been done by the EOS community on solutions to the full history. This project will draw on the work done by other teams in order to mitigate project risks. Notable references are;

1. EOSIO RPC API developers reference, https://developers.eos.io/eosio-nodeos/reference

   This reference defines the history API endpoints.

2. EOSIO history plugin, https://github.com/EOSIO/eos/tree/master/plugins/history_plugin

   Original EOS history plugin.

3. EOSIO history API plugin, https://github.com/EOSIO/eos/tree/master/plugins/history_api_plugin

   Original EOSIO history API plugin.

4. EOSIO MongoDB plugin, https://github.com/EOSIO/eos/tree/master/plugins/mongo_db_plugin

   The EOSIO MongoDB plugin which was created to replace the (deprecated) history and history API plugins.

5. Greymass hapi branch of history plugin, https://github.com/greymass/eos/tree/hapi-production/plugins/history_plugin

   Modifications to the EOSIO history plugin which resolve shortcomings in the original EOSIO history plugin by splitting the history file into multiple smaller files. This allows the history plugin to continue working on the EOS mainnet.

6. EOS Lao Mao Elasticsearch plugin, https://github.com/EOSLaoMao/elasticsearch_plugin

   An alternative “DB” plugin by EOS Lao Mao which allows history records to be stored in Elasticsearch instead of MongoDB. Benchmarks show approximately 250% performance improvements over MongoDB.

7. Atticlab Elasticsearch history API, https://github.com/atticlab/eos-es-historyapi

   An EOS history API by Atticlab based on Elasticsearch and built using GO. Something similar will be required to deliver history API services on Cassandra.

8. Public work proposals for EOS infrastructure, EOS Chronicle Project, https://github.com/EOSChronicleProject/eos-chronicle
9. https://github.com/cc32d9/eos_zmq_plugin_receiver
10. https://github.com/cc32d9/eos_zmq_plugin

History API system architecture

Nodeos running Cassandra plugin

A node running nodeos with the Cassandra plugin is necessary to load blockchain transactions into the database. This is identical to the way the MongoDB history solution works. The workload for this node is limited by the transaction rate of the blockchain. Because transaction processing on the blockchain is a serial process (single threaded) we will not see massive growth in load on these nodes. Scaling at this point is not a significant risk.

A production system would probably require a failover node running the Cassandra plugin to provide failure resilience. This is an architectural consideration when implementing this history solution.

Cassandra cluster

The Cassandra cluster provides a storage solution that is able to scale horizontally as the requirement for data storage grows.

Web servers servicing API endpoints

History API endpoints are served by a web server which queries the Cassandra cluster. The biggest scaling issue for the history API is the ability to service web requests. As blockchain utilization grows the number of web requests will grow.

The web server may connect to any Cassandra node to service the query request.

Estimates value of 28.3M votes

Revenue from the CVIB would be used to support infrastructure and the team. In addition some of the funding may be needed to reward outside consultants where the necessary skills do not exist within our teams.

There is a risk that the vote based rewards may fall below the EOS minimum daily reward of 100 EOS. This is, however, only a risk if the BP does not have previous votes. EOS Nairobi is already a paid standby BP. The risk of falling below the minimum daily threshold is therefore greatly reduced because of other votes supporting the BP.

Costs

Development / minimum viable product

For development purposes there is a choice to develop on bare metal or host on a cloud service. A cost comparison is needed to establish the best option. The bare metal option is presented here. The estimated hardware cost of this approach is approximately US$3,000 (835 EOS @ $3.59).

Jungle testnet

Hosting

Bare metal, own infrastructure

Hardware

Nodeos server

CPU : Intel i7
Ram : 32GB
Disk : SSD 500GB
Cost : US$1,200

Cassandra cluster (1, 2)

(3) 5 x Cassandra nodes
Redundancy factor : (?) 3
Docker host : Core i9-9900K, 8 Core CPU, (8) 16GB Ram, 500GB Disk
Docker containers (each) : ? CPU, 2GB Ram, 80TB Disk
Cost : US$1,500

Web server

Docker host : (Host on same machine as Cassandra)
Docker container : ? CPU, ?GB Ram, ?TB Disk

Notes:

1. Docker Docs, Cassandra, https://docs.docker.com/samples/library/cassandra/
2. Build A Cassandra Cluster On Docker, Gokhan Atil, https://gokhanatil.com/2018/02/build-a-cassandra-cluster-on-docker.html

Production

The costs to operate a production API is based on the following assumptions.

Blockchain

EOS Mainnet

Hosting

Data center / cloud,
Bandwidth : 2 TB/day (1)

Hardware

Nodeos server

Cassandra cluster

Web server

Notes:

1. Based on posting by sw/eden on CVIB Telegram group (https://t.me/CVIBEOS/501).

Delivery timelines

Phase 1

The primary objective for phase 1 of this project is to build a horizontally scalable history database and prove the robustness of solution.

R&D and development environment — 3 weeks

- Commission development environment hardware
- Software installs (Nodeos, Cassandra, web server)
- Development system access (Firewall, VPS)

Cassandra schema — 1 weeks

Cassandra does not support table joins. The schema needs to be able to support API endpoints without the need for table joins.

Load end-to-end — 2 weeks

Build the Cassandra loaded and load blockchain data into Cassandra

API web service — 2 weeks

Create web service to replicate the following RPC endpoints.

- get_actions
- get_transaction
- get_key_accounts
- get_controlled_accounts

Validation and open community testing, operational testing — 4+ weeks / ongoing

- Validate the API against current history alternatives to prove the consistency and correctness of the data.
- At this point the API can be opened to community developers who want to evaluate the solution performance.
- Test the addition of Cassandra nodes, retirement of Cassandra nodes and node failures under continual operation.

Phase 2

Preparation of system for production. This phase will need to be done in a hosting environment that is not subject to bandwidth limitations of phase 1. A budgeting exercise will need to be conducted before proceeding with this stage.

Production system design — 3 weeks

This involves completing the system (infrastructure) design for a full scale mainnet implementation of the solution.

Nodeos plugin optimisation — 2 weeks

Fix and test any nodeos plugins identified during phase 1

Web API optimisation and scaling — 2 weeks

Configure and optimize multiple API nodes behind a load balancer.

EOS Mainnet synchronization — 4 weeks

Synchronize the solution with the EOS mainnet and make available.

Full scale benchmarking / load testing — 2 weeks

Carry out benchmarking / load testing to prove the suitability of the solution for the needs of the community.

Collaboration

This collaboration initiative is being undertaken be EOS Nairobi and EOSza. Both of these entities are actively involved in growing the adoption of EOSIO on the African continent.

EOS Nairobi

EOS Nairobi was founded by Daniel Kimotho, Felix Macharia, Steve Kiarie, and Brian Kimotho. EOS Nairobi is contributing to the geographic distribution of the network by hosting bare-metal infrastructure in Nairobi, Kenya. With a focus on training African developers and incubating the entrepreneurial spirit in Africa, EOS Nairobi shows that their focus is on planting the seeds necessary to grow EOSIO on the continent. EOS Nairobi has partnered with OCI and Phil Mesnier to run EOS developer workshops, while also collaborating with various universities in Africa towards the same end. The team has also incubated and advised various EOS startups in Africa such as Ubuntu Energy Ledger, Africonnect, and Team Btax. Members of the team are also founding members of the Africa Digital Asset Foundation (ADAF) which is working to create a dynamic framework for standardizing digital assets from a technical and legal perspective across the continent of Africa.EOS Nairobi is actively involved on a number of EOSIO based blockchains and an active member of the EOSIO community. Notable activities include;

- Paid EOS mainnet standby producer.
- Paid Telos mainnet standby producer.
- Paid Worbli mainnet standby producer.
- Active Africa developer community development
- Participants in EOS London and African Global Hackathon events.

Team

Daniel Kimotho

Co-Founder of EOS Nairobi.

Daniel has a degree in Business Information Technology from Strathmore University.He is an affiliate researcher with the Institute for Blockchain Studies (New York)).Daniel is also co-founder of First Nexus Company, a technology company focusing on data and emerging technologies for enterprises.Daniel is also involved open Finance and open Finance Governance.

Felix Macharia

Co-Founder of EOS Nairobi, Head of Operations.
Felix has a background in scientific research and entrepreneurship. He is a co-founder at First Nexus Company, an innovative technology company that focuses on developing solutions for business that take advantage of data and emerging technologies.
He is also an affiliate researcher for The Institute for Blockchain Studies (New York).
Felix has held several operations positions in previous organizations including: Lighta Africa (Operations and Public Relations Manager, 2012-2014), Omnis Limited (Business Development Manager, 2015-2018), Connect Wild Africa (Head of Research and Strategy).
He is passionate about the real-world use cases of blockchain technology and sees it as the greatest enabler of financial inclusion

Steve Kiarie

Head of Technology EOS Nairobi
Steve is the lead server architecture specialist in charge of block-production.
He has a background in Network Security and Server management with his consulting firm that has been in operation for more than 8yrs, serving clients in Africa and Europe.
Steve is currently working on implementation of Smart contracts and Decentralized financial systems as well as offering Cloud based Server and Backup Solutions to local organizations.

EOSza

The founders of EOSza have 20 years experience in designing and building big data solutions for call records for the telecommunications industry. These solutions allowed simultaneous load and query into petabyte scale databases at transaction volumes of 5 billion+ transactions per day. This experience with high performance production systems positions us ideally to provide similar solutions to the EOSIO community.

EOSza is actively involved on Telos and EOS mainnets and an active member of the EOSIO community. Notable activities include;

- Paid Telos mainnet standby producer.
- Unpaid EOS mainnet standby producer.
- Mentors at inaugural EOS Global Hackathon, Hong Kong.
- Participants in EOS London and African Global Hackathon events.

Team

Rory Mapstone

Founder of EOSza, formally founder of LGR Telecommunications.

Rory holds a masters degree in mechanical engineering. He began his working career in 1990 as a design engineer specializing in thermodynamics and computational flow. He was one of the founders and chief technical officer of LGR Telecommunications, a company that specialized in developing software and big data solutions for the telecommunications industry.

Rory began working with EOSIO in January 2018 when he successfully completed a proof of concept porting internal projects to EOSIO and became convinced that the blockchain could be applied to real business problems.

Rory co-founded EOSza following the sale of LGR Telecommunications. He attended the EOS Hong Kong global hackathon event as a mentor and participated in the London and Africa event.

Luck Fatsilidis

Founder of EOSza, formally founder of LGR Telecommunications

Lucky holds a degree in computer science. He began his working career as a telco switch software developer at Siemens and then as software developer at MTN mobile network. At both these companies he designed and built software solutions that were later sold commercially.

He was one of the co-founders and chief operating officer of LGR Telecommunications. He authored LGR’s flagship product, CDRlive which has been used widely by mobile telecoms operators to analyze call records.

Lucky co-founded EOSza following the sale of LGR Telecommunications. He participated in the EOS global hackathon events in London and Africa.

Andreas Evangelou

Founder EOSza

Andreas holds a degree in computer engineering. He joined LGR Telecommunications as a software developer. He became a co-founder of EOSza following the sale of LGR Telecommunications in mid 2018.

Andreas attended the EOS Hong Kong global hackathon event as a mentor and participated in the London and Africa event.