

Blockchain Projects BV

Zutphenseweg 6 7418 AJ Deventer The Netherlands

Chamber of Commerce: 697 131 38

www.blockchainprojectsbv.com info@blockchainprojectsbv.com

Roadmap for worker 2017-08-infrastructure 25.07.2018

Milestones Deliverables:

- 1. Maintain deployed infrastructure from previous worker According to description and constraints of previous infrastructure worker
- 2. Deploy public uptime tracking
 - a. Evaluation of different possibilities (custom uptime robot, metricbeat, ...)
 - b. Deployment in alignment with the existing stats pages
- 3. Switching to premium SSL certificates for all infrastructure nodes
 - a. Obtain Premium SSL certificates for every subdomain (not one wildcard)
- 4. Setup of ES databases, cluster configuration and feeding through bitshares-core
 - a. Configure every backend node to feed the ES cluster (pending completion by core development team)
 - b. Direct access to ES cluster (read-only) through

elasticsearch.bitshares.ws elasticsearch.testnet.bitshares.ws

c. Access to ES python wrapper through

wrapper.elasticsearch.bitshares.ws Wrapper.elasticsearch.testnet.bitshares.ws

Please note that for point b. and c. it might be necessary to deploy the subdomains [eulus|sq] to avoid unnecessary traffic costs, this will be decided during deployment. The ES database for the testnet will not be a cluster.

- 5. Development of automated deployment and execution of stress-testing
 - a. Stress beacon

- i. Develop a light-weight python that is remote controllable. Remote features include (no guarantee for completeness)
 - 1. Configure the BitShares accounts to be involved in the transactions and backend node to be used
 - Create and sign transaction for later broadcasting that contain an arbitrary amount and random operations. Amount of operations per transaction and distribution of the operation type is configurable
 - 3. Broadcast previously signed transaction to configured node at a given tx/s rate (can be unlimited)
 - 4. Report on client-side statistics: server CPU load, failed broadcasts, connectivity issues. This can be either directly back to the overseer or to a third monitoring entity
- ii. Create a docker container for easy deployment
- b. Stress overseer

Develop a python frontend that allows managing and control of all beacons

- 6. Development of proper frontend for monitoring stress testing
 - a. Deploy Kibana for the ES testnet monitoring
- 7. Manage, execute the stress test
 - a. Deploy multiple stress beacons and one overseer, and if required additional testnet nodes
 - b. Execute various scenaria
 - i. Real user activity

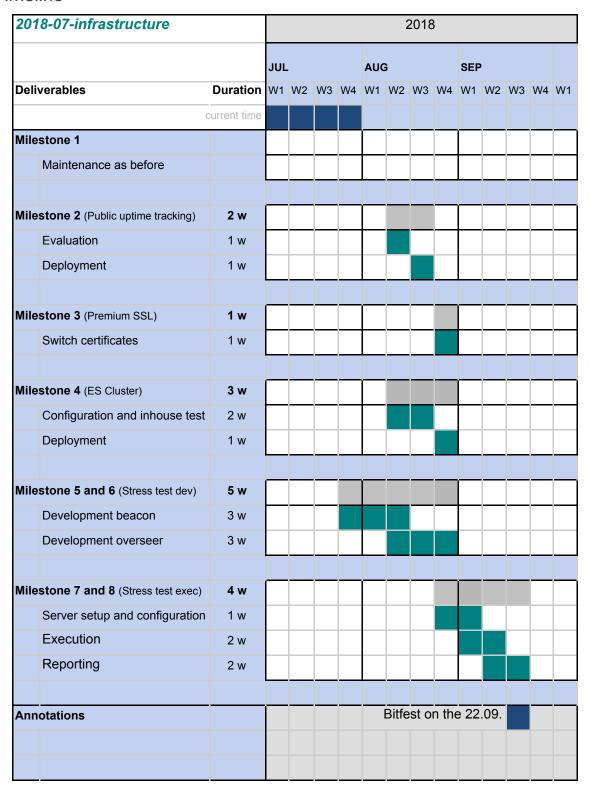
Most likely only one operation per transaction, incoming transactions more or less distributed equally accross nodes, mostly transfer and limit order

- ii. Maximum throughput
 - 1. Transactions with single transfer incoming on multiple nodes
 - 2. Transactions with single limit orders incoming on multiple nodes
 - 3. Transactions with single transfer incoming on only one node
 - 4. Transactions with many transfers incoming on multiple nodes
- iii. P2P limit (might be already proven with maximum throughput)

Transactions with so many operations to reach the 2MB limit incoming on multiple nodes

8. Create in-depth stress test report

Timeline



Please note that the durations are merely giving indicating the time period in which the underlying deliverable. It is not indicating the required workload.