Notes (in progress) about P2P Web

Rasmus Erik Voel Jensen

2017

Abstract

Random ramblings and notes during the P2P Web.

Presentation notes

Purpose:

- Infrastructure for no-server HTML5 apps => a decentralized trustless computer for the web

In short:

- Network topology: kademlia like, - address = hash of pubkey
- State/storage: each node stores a neighbourhood around its own address, saved in blockchain merkeltree
- Operations: changes to state are verifiable, and verified by nodes in neighbourhood
- Balance: nodes gets paid for doing tasks for the network, and can use this to buy tasks in the network. Also pay/payout for state blockchain.
- Tasks: stored, nodes assigned to tasks in deterministic random part of storage, proof-of-result stored, result stored, verification/value, balance updated.

Additional notes:

- WebPlatform: computations in webassembly. WebRTC as transport (thus modified kademlia). Crypto-algorithms from crypto.subtle.
- Neighbourhood size and amount of state per node - determined by node density (global minimum density / local density). Fixed amount of memory per node.
- Mutable references in blockchain (using balance to keep alive)
- Autonomous processes (using balance to keep alive)
- Not entire blockchain stored, only parts needed by the node

- Stake in computation tasks
- Balance/trade between processes
- Introduced 'errors' in blockchain, and bounties for finding/proving them.
- Binary/Quad merkel tree for proofs
- Pub/private-key derived from entropy source
- Task types: computation, storage, storage-transfer, find node with certain data
- Node trust / reliability proof via blockchain
- Block-tree rather than chain
- Computational task level of validation
- Result safety: added to state by any node in neighbourhood by proof of distance of computing-node to task.
- Computational task: computing time bound, and cost calculation.
- consensus algorithm: CRDT, additional data in timeinterval: after last block, before timed signature from other deterministic random node
- Tagged overlay network - opt-in part of infrastructure for tunable bandwidth requirements
- Network simulation (core optimised for low memory)
- Bandwidth optimised, - number of significant bits per node-id, stream compression, only send diffs etc.

Explore/ideas:

- Performance characteristics of current WebRTC implementations
- Performance effects of design choices for Kademlia-like algorithm on top of WebRTC (instead of UDP)
- Verifiable "computational" tasks, and economy based "computation".
- (Survey p2p overlay networks)
- WebRTC bootstrapping options (decentralised signalling server vs. actual node)
- Infrastructure deployment - bootstrap-code + load signed version of code from network, - partly test within network before full deploy.

Description of algorithm:

- nodes connected in kademlia-like structure
- regular state snapshot (blockchain merkel-dag)
  - divide-and-conqueor consensus algorithm, verifying credit updates in neighbourhood.
  - each node stores the state of a neighbourhood around its own address, as well as the path to the root. The neighbourhood size is fixed for all, ensuring good redundancy of data for
- content of state
  - list of entities(nodes)

        * id
        * balance/credits (updated by work, tasks, cost of staying in blockchain, and transfers)
        * state (+ proof)
        * tasks - scheduled for execution - wager
        * result of previous scheduled task
        * work
            · stake
            · result
            · proof-of-work
        * state
- verifiable tasks
- task types
    - computation
    - storage
        * data
        * key/value
    - random verifications (of proof-of-stake tasks)
    - blockchain verification
- entities
    - nodes
    - nodes with stake
    - accounts (pub-key)
    - autonomous
- computational process
    - task gets stored in blockchain
    - task gets assigned to a number of bcrandom nodes
    - task gets done, and proof-of-work gets stored in the blockchain
    - task result gets released
    - result+proof-of-work get validated + signed into blockchain
    - balance is updated

Design criteria

- low bandwidth
- low memory footprint (useful for large simulation, as well as embeded systems)
- low code footprint
- tagging of hosts
- connect to arbitrary host
- foundation for other p2p applications

Brainstorm around potential articles

Possibly relevant conferences or journals

Relevant Litterature

- (Parno, Raykova, and Vaikuntanathan 2012)
- (Kaune et al. 2008)
- (Wood 2014)
- (20ADa)
- (20ADb)
- (20ADc)
- (20ADd)
- (20ADe)
- (20ADf)
- (20ADg)
- (20ADh)
- (20ADi)
- (20ADj)
- (20ADk)
- (20ADl)
- (20ADm)
- (Ben L. Titzer Michael Holman Dan Gohman Luke Wagner Alon Zakai JF Bastien 2017)

https://allquantor.at/blockchainbib/bibtex.html

Notes for later / optimised version

- page size
- typically 4K (getpagesize() is 4K on my linux, and that looks like common size via https://en.wikipedia.org/wiki/Page_%28computer_memory%29)
- webassembly 64K page size
- minimise memory usage (for ability to run large simulations).
- i.e. 64K per nodes => 100K nodes in memory simulation ~ 6G memory

Bibliography

Ben L. Titzer Michael Holman Dan Gohman Luke Wagner Alon Zakai JF Bastien, Andreas Haas Andreas Rossberg Derek L. Schuff ∗. 2017. "Bringing the Web up to Speed with Webassembly."

Kaune, S., T. Lauinger, A. Kovacevic, and K. Pussep. 2008. "Embracing the Peer Next Door: Proximity in Kademlia." In *2008 Eighth International Conference on Peer-to-Peer Computing*, 343–50. doi:10.1109/P2P.2008.36.

Parno, Bryan, Mariana Raykova, and Vinod Vaikuntanathan. 2012. "How to Delegate and Verify in Public: Verifiable Computation from Attribute-Based Encryption." In *Theory of Cryptography: 9th Theory of Cryptography Conference, Tcc 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings*, edited by Ronald Cramer, 422–39. Berlin, Heidelberg: Springer Berlin Heidelberg. doi:10.1007/978-3-642-28914-9_24.

Wood, Gavin. 2014. "Ethereum: A Secure Decentralised Generalised Transaction Ledger." *Ethereum Project Yellow Paper*. http://bitcoinaffiliatelist.com/wp-content/uploads/ethereum.pdf. http://bitcoinaffiliatelist.com/wp-content/uploads/ethereum.pdf.

20ADa.

20ADb.

20ADc.

20ADd.

20ADe.

20ADf.

20ADg.

20ADh.

20ADi.

20ADj.

20ADk.

20ADl.

20ADm.