Quantum1Net Blockchain
“Yggdrasil”

Architectural overview

Thomas Olofsson, Mattias Bergström

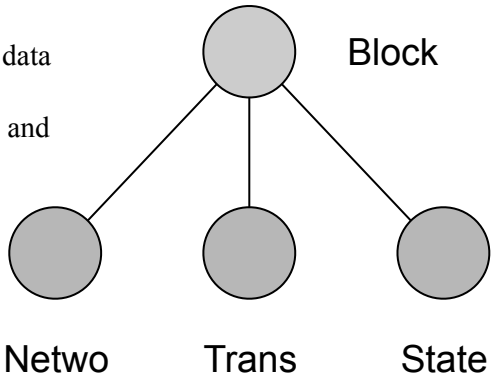
Background

This document is a design document for the blockchain part of the quantum1net network. The quantum1net network is a quantum-safe overlay network over the internet not unlike a routed vpn network. The network is controlled by the the quantum1net coin through the blockchain implementation yggdrasil as described in this paper.

Overall architecture

While most current generation blockchains is implementing a peer to peer network with the sole purpose of supporting transactions Quantum1net is seeking to extend this.

The Q1N network is supporting transactions of any encrypted data directly between the nodes or relayed over the network over several nodes. These transactions are cryptographically secure and is reported to the blockchain by the nodes.



The overall architecture of the chain is loosely based on second generation chain technologies as in that the a global state is shared amongst all nodes on the network. Building upon that, the chain is also using network performance as a measurement of work for the proof of work validation.

On the architecture of the blocks

Just like Yggdrasil the tree of life in Norse mythology, Quantum1Net Yggdrasil has three roots. Each Yggdrasil block has three roots, these roots are represented as root_hashes

Transaction root

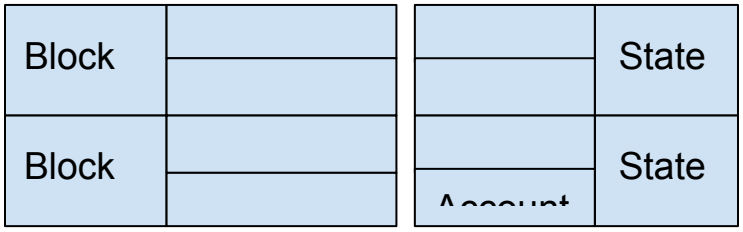
This is the root of all transactions in the block. This is similar to the transaction tree as defined in bitcoin and other first generation block chains. The chain is committed each block and all transactions are saved in the chain.

State root

This is the root of the state of the accounts in the network as well as the state of the currently used keys of each account. The state has the information of all accounts in the blockchain, it is not stored in each block. The state is generated processing each block since the root block. Each block will only modify parts of the state. In addition to the state of the account here is also stored the nonce of the account to protect against double spending transactions. The state root is also responsible for tracking any code associated with an account in the case of smart contracts and is stored in a storage tree (trie) in memory on the nodes.

On keeping state and security of the accounts

In the Q1C The state of all accounts in the chain is calculated from rerunning all transaction on all previous blocks in the chain starting with the genesis block. This means that the state information is never transmitted over the network and this makes the data needed to be transferred for each



transaction about half the size of a normal bitcoin transaction.
In each minted block a hash of the state tree is saved. The state of all transactions is stored in a “trie” structure which is then used to validate the state for that block height.

Network root:

This is the root of the network status messages needed to calculate the network state. Each participating node broadcasts their contributed work to the peers in the network once per block. This is stored in a trie tree structure (Trie) similar to the node state and is ordered by node-id (wallet address). For each block this data is validated by the minting node and the verifying nodes. The signed data in the previous block is used to calculate and track the committed proof of contribution to the network. Information in this root is also used for routing of the overlay network and can be seen as a map of the network as well as being used as a input and proof of contribution for the consensus of the network.

Blocks and block signing

To achieve a highly efficient and scalable blockchain solution the overall architecture must support some way of parallelising the signing of transactions and consensus implementation.

Decentralised Regional block signing and master blocks

To achieve low latency transaction verification the network can perform sharded signing. This will allow the network to pre-process the transaction with a high but not total certainty before committing the transactions to the master chain. The formation of local clusters is made by using the network state of the chain to see which nodes are located close to each other from a network perspective.

Transactions can state whether they require full master block signing or regional only signing to accept the transaction as completed. If both parties in a transactions are in the same regional hub the state of the accounts will be updated quicker on that subsection of the network in a the time of a regional block. The regional blocks will eventually get synced to the main chain in a upcoming master block.

A regional cluster can send transactions faster due to network and geolocation optimisations and will have a block time of circa one second in full production. Every master node that is assembling or verifying a block shall take a list of hashes of all processed and signed regional blocks and append that to a separate hash tree in the block. This way the local blockchains will be tied to the master blocks every 20 seconds but transactions can happen faster ideally sub-2-second on regional signing if both parties agree to regional signing.

This means that there is a potential window for double-spending if two transaction from the same account appears in two regional clusters and are regionally signed before being committed to the master state. In this case, the transaction with the lowest timestamp will be accepted as correct and the second transaction with the same nonce will be rolled back from the nodes state.

Quantum safe Transaction signing and block signing

Each master and regional block shall be signed by a quantum proof xmss signature from the signing nodes keyring.

Each transaction is signed with a non-quantum safe ecdsa key. The reasoning for choosing ECDSA is that the security of the key in relation to signature length is unprecedented. To secure the transactions the client will update the account on state with a newly generated key every time the client signs a transaction. This way the transaction key will only be vulnerable to attack until the transaction is confirmed in a block this limits the attack time to 20 seconds and even with quantum computers, these attacks will not be feasible.

This results in a high level of post quantum security while still maintaining a low block size and high transaction speed in the network

Block size and block times

The block time is initially set to 20 seconds per block. It is estimated that after optimisations the block time should be able to be reduced to 10 seconds.
The block size of a Quantum1Coin block is dynamic and is limited by either a maximum of transactions (200,000) or a maximum transaction fee per block, giving a theoretical throughput of 10k trans/s. It is estimated that after optimisations the block time should be able to be reduced to 10 seconds.

Network and peer to peer

All nodes in the network are connected through a full mesh to as many nodes as possible. Each node joining the network will do a key quantum secure key exchange and set up a session key for each tunnel in the mesh. The key exchange is protected by the xmss keypair of the node. This way each node is also securely authenticated to the network and can be identified by the node-id which is also the wallet address of the node.

The steps to run a block in the network are as follows:

1) New transactions are broadcast to all valid nodes in the VNL through secure tunnels.
2) Each node collects new transactions into a open-block.
3) The n number of nodes selected by the consensus algorithm to mint the block works on validating the transactions.
4) When a node finds a solution to the block including all network entries it reports it to the validators.
5) Nodes accept the block only if all transactions in it are valid and not already spent.
6) Nodes express their acceptance of the block by working on creating the next block in the chain, using the hash of the accepted block as the previous hash.

Transaction types

In the chain, there are four main types of transactions. These are:

- Monetary transaction
- Network transaction
- Network reports
- BinB transaction

Monetary transactions

This is the base transactions to move coins from one account to another. Similar to a normal transfer of coins and tokens in most cryptocurrencies. Monetary transactions do also initiate a key change for the account

Network transactions

This is a transaction in which a user pays for and set up an encrypted tunnel to any node in the network. Before the network transaction is sent to the blockchain the two nodes will negotiate a fee. On top of the normal monetary transaction, the data portion of the transaction is filled by the client with a challenge to be used to set up the session key for the session. This is then countersigned by the server and then signed by the minter.

Network reports

To be eligible for minting each node need to prove their contribution to the network. For each block, all participating nodes in the chain shall submit a network report containing how much network traffic they have processed in the network. This will be added to the blocks network root and is stored in the blockchain. This will cause a slight block size overhead as compared to other blockchains but the overhead will only be 50 bytes times the number of full nodes in the network.

Blockchain in Blockchain transactions

Biib transactions Are private of public data sets that are loosely connected to the main chain every n number of blocks from the main transaction chain to facilitate logs and receipts and is intended to be handled by separate singing servers that reports the completed blocks to the master nodes. data sets or blockchains, signed by a service holder and used to support their specific service.

For instance, a service provider on the network can choose to only sign transactions for a specific chain id. The network fees for bib transactions are 100th of the normal network fee.Transaction fees and transaction priority
All transactions need to have a minimum transaction fee to make it into a block. The minimum transaction fees are configurable but will initially be set to 100 Baldr for a monetary transaction and 1 Baldr for a BinB transaction. As for Network transactions, the fees are the same as in monetary transactions.

Decentralised Consensus and Minting

To implement a distributed timestamp of signing Bitcoin introduced a proof of work algorithm based on hashcash[1] to decide who is signing the next block in the chain. This comes with several problems where the most obvious problem is the extensive power consumption of the bitcoin transactions[2] [3] as well as the problem with increased transaction times as the complexity of the proof of work increases.

Since any peer to peer networks efficiency and reliability is based on the data that the network is able to safely and successfully relay between the peers we propose a solution to the distribution problem by measuring network throughput of the participating nodes of the network.

Quantum1 net is based on a model called proof of transfer where each node tells the network its network utilisation on a periodic basis and commits this information to the blockchain. This information can then be used when signing a block to make a tally of which nodes that are contributing most to the network and thereby increasing the chance to win the right to sign the next block. This methodology is combined with a proof of stake methodology to counteract spoofing of network info transactions by a malicious node.

Each node is denominated by the wallet address and need to have a minimum stake of n coins to be included in the network.
An additional security by proof of stake supports the network in detecting fraudulent nodes. An attacker found by the network would be excluded from the consensus mechanism and could on repeated offenses lose the coins stake which would then be redistributed over the network.

Trusted Nodes

All nodes that are relaying data in the network sends status updates every block. Measuring how much data they have sent and received as well as ingress and egress to the network.
These transactions are called Stats transactions and are a part of the “Proof of contributing work”.
To be eligible to be part of minting and count as a trusted node a node needs to have committed work into a minimum of nine of the 10 last blocks.

Minting remuneration

The Yggdrasil blockchain implements the concept of minting. The reason for this is to solve the decentralised consensus in a green and efficient manner as compared to the normally applied proof of work.
Each of the trusted nodes that get elected to mint a block gets the opportunity to assemble a block and add its own coinbase transaction to the root of the block.

---

[1] **Hashcash - A Denial of Service Counter-Measure Adam Back**

[2] **http://karlodwyer.com/publications/pdf/bitcoin_KJOD_2014.pdf**

[3] **https://www.researchgate.net/publication/322118225_Bitcoin_miners_true_energy_consumption**

Each new block with a minimum required transaction fee sum generates a coinbase transaction with the value of the sum of the transaction fees for the block as well as a minting bonus based on a logarithmic scale proportional to the the total number of bytes contributed to the block.

On minting selection

In the q1n chain, three minter's are selected through a pseudo random process with a proportional distribution of minter success proportional to the network transmission they have provided in the last minted block. This way running nodes in the network will be rewarded by coins similar to traditional mining but instead of contributing hash power the nodes contribute network transaction power.

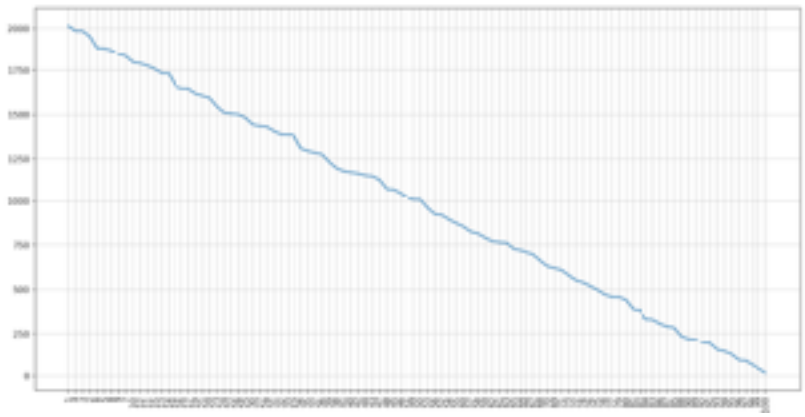
The three minters who gets the right to mint the next block are calculated by all nodes based on the previous block by the following formula.

$$K \bmod\left(\sum_{i=0}^{n}(t_i + r_i)\right)$$

Where K is a random number bigger than the total amount of network traffic, t is transmitted data and r is received data.
Each node has an allocated range slot in the network reward table proportional to the contributed network traffic

The Total block transfer volume is calculated by adding all transfers from all nodes in the block
Every node is given a slice of opportunity by dividing the contributed work of the node by the sum of all contributed work of the block.

The minter is selected by converting the root hashes of the previous block to a large integer and taking the modulus of the sum of the total network traffic generated by the block. This has been proved in simulations of 100.000 blocks to yield a frequency distribution that is proportional to the work put in.



Valid block verification

All verifying nodes in the network verifies the incoming signed blocks signatures and verifies that the accepted block is minted by one of the elected minters. The block with the lowest timestamp and the correct block-height is accepted as the winning block. The two blocks with higher timestamps is be called supporting blocks and will also yield a 5% fee in the next block. (supporting block fee).

Detection of fraudulent nodes and blocks

By tallying up the claimed proof of work by the nodes every node also keeps a record of how much data it has received from each peered node in the network. If a node sees a discrepancy of the reported work by any other node and the network traffic seen as part of the whole traffic the node should make a proposal to vote the node off the network. If a majority of peers in the network sees a similar discrepancy the node is kicked out the network and the tokens staked are added to the coinbase transaction of the next block.

The proof of stake supports the proposed solution solves the problem of representation of nodes in the majority decision making. Each node is denominated by the wallet address and need to have a minimum stake of n coins to be included in the network.

Network Nodes

Full nodes

The full nodes are nodes that are fully meshed into the core network and can participate in the transaction signing and minting.

Client nodes and Oracle nodes

To enable a more high speed set up of full mesh node and to limit the memory size and to enable purging of inactive accounts it is proposed that a qualified majority of full nodes could and should vote to make a known good baseline at a block height equivalent to 12 months.

This means that anyone that sets up a minting node only will have to download the transactions for the last 12 months and calculate the state of all nodes from there.

This will greatly increase the speed for the vast majority of blocks and cull inactive accounts from the majority of nodes. Only the (hopefully) rare occasions when an inactive account reemerges in a block will the full node signing the block have to request the state from the Oracle node.

If an account is seen in a transaction that is not found in the full nodes state the node shall then ask one of the Oracle nodes about the state of that account. There must be a minimum of n (yet to be decided) Oracle nodes in the network that keeps all transaction information from block zero and forward. These nodes are to be seen as supporting nodes to the network and any user of the network are welcome to set up additional oracle nodes.

A oracle node can be any node that is participating in the network but since the requirements of keeping the entire chain and state in cache the hardware requirements for these nodes are considerably higher than the requirements for a normal minting node.

Threats and countermeasures

Here we describe some of the most common security problems associated with blockchain

Double spending

Double spending is where an attacker uses the same previously spent transaction by using the output of a previous transaction.
To protect against double spending Each account has a globally accessible nonce which prevents same-chain replay attacks and double spends.

The nonce is the sequence number, which minters check because a block that has a transaction with an incorrect nonce is an invalid block (other minters or nodes won't build on top of it). This is a proven way to counter double spend by second-generation blockchains and is described in depth in several papers

51% attacks

The 51% attack which targets a majority consensus network gives the ability of someone controlling a majority of the network hash rate to revise transaction history and prevent new transactions from confirming.

Since the Q1 network do not use a majority voting for signed blocks but rather a pseudo random function hijacking and splitting the network is harder than maintaining 51% of the nodes since theoretically any node on the network could become the block winner. To protect against network splits of malicious actors a proof of stake is used. Nodes that are detected forging or minting fraudulent and non valid blocks can be detected by the peers on the network and will be invalidated from signing new blocks and can also lose the staked coins in the associated wallet. This makes a network takeovers harder than in a majority based system.

Sybil attacks

In a Sybil attack, the attacker subverts the reputation system of a peer-to-peer network by creating a large number of pseudonymous identities, using them to gain a disproportionately large influence. A reputation system's vulnerability to a Sybil attack depends on how cheaply identities can be generated, the degree to which the reputation system accepts inputs from entities that do not have a chain of trust linking them to a trusted entity, and whether the reputation system treats all entities identically.

Since all nodes on the network are identified by a account in the global state that have a associated key the system is inherently resistant to sibyl attacks. The addition of proof of stake in the consensus mechanism further protects the network against this type of attack.