Well, times have changed. And it has happened quite quickly. Several years ago lots of Bitcoin early followers had difficulties with their funds, but they held onto their beliefs and their coins, thuswise they are feeling like nobody's business now,- the experts in Crypto Industry say, - we still need Bitcoin and Ethereum to work on a larger scale, so businesses need to decentralize data and ensure their privacy concerns. Now we face a new challenge: considering the huge amount of money invested, it remains to be seen how many old-timers and newcomers will remain faithful to the industry and will strive to change the world with the help of technologies that have already changed them.

Nowadays, Blockchain technology is experiencing a huge boom. New Blockchain schemes are created daily, including the largest technical corporations. For example, Microsoft offers his clients tools for experimenting with crypto currency in Azure cloud. IBM, Intel and so forth are collaborating with the Hyperledger hub, an open platform for developing business-centric Blockchains. The major banks, the very ones that the creators of the cryptocurrency wanted to oust, came up with their version of technology, trying to outrun the trends.

Even Bitcoin, which works on the premier and the most successful Blockchain, is being upgraded for apps whereof its creators have never dreamed of.

Nevertheless, there has already appeared a potential funding mechanism for the Blockchains - Initial coin offering or ICO. It turned out to be extremely lucrative, although legally dubious.

For example, a person decided to use the app, but he cannot make payment using common currency, he needs to buy special coins for this app, which were previously released to the market, and pay them off. In the real world, it would work like this: someone has opened the laundry and issued tickets, which can be paid for washing. The owner beforehand sells all tickets to people, and then they resell them to others, if it's necessary.

Currently, more than half a billion dollars invested in the tokens sale and recently these numbers are only growing. For example, in July the Tezos Blockchain set a record collecting more than $ 200 million through the ICO.

Blockchain technology can be useful not only for making transactions. Almost straightly after Bitcoin emerging, people began to think about how to use this technology in other fields. When miners check the transactions, they run small programs that process and provide the data necessary for the transaction. But what if we run more complex programs? Software for social networks? Or use Blockchain to provide data for online forums?

These ideas emerged right along after the creation of Bitcoin, but only a few years later a nineteen-year-old student from Toronto contributed to their development. In 2013 Vitalik Buterin developed a completely new technology called Ethereum. Because of Ethereum Blockchain can be used not only for making transactions.

Unlike Bitcoin, Ethereum uses mini-programs called smart contracts, which can be written with an unlimited level of complexity. Users can interact with programs by sending them guided transactions that are then processed by the miners.

It means that everyone can embed a program in transaction and be sure that it will remain unchanged and accessible to the Blockchain. Hypothetically Ethereum can replace Facebook, Twitter, Uber and any other digital service with new versions that are transparent, invulnerable to censors and do not require human intervention.

1. Project Background

Recently the Ethereum mark-up and the increased attention to the platform has led to a serious escalation in the cost of Ethereum based decentralized apps and increasingly expensive usage of them. At the same time, the demand for these apps is superlative and there is no reason to believe that this trend is going to change in the nearest future.

All that happen because in the Ethereum network users must pay directly for the compute capacity they use regardless of whether they have made a bet on the market of predictions or they exchange messages using a decentralized messenger. This is slightly different from using Facebook, which is a shareware platform.

Most recently, a screenshot has been published where we can see the process of creating a user profile in one of the apps and it costs 0.08 ETH or $7. “It cost a little over than $1, when the Ethereum cost $10”, wrote one of the cofounders of this app.

Considering this growth, many leaders of the decentralized start-ups believe that “It's easy-to-understand that some users can get mad”.

GasPrice

However, a great many Ethereum apps are now in the testing phase. Therefore, it remains obscured how many apps and users suffered from the Ethereum price increasing at the moment. Therefore, this situation can cause certain problems in future.

In fact, everything is a tad trickier. Gas price assessment is dynamic and depends not only on the cryptocurrency value. It stands to mention that “gas”, “gas price” and “gas cost” are different terms.

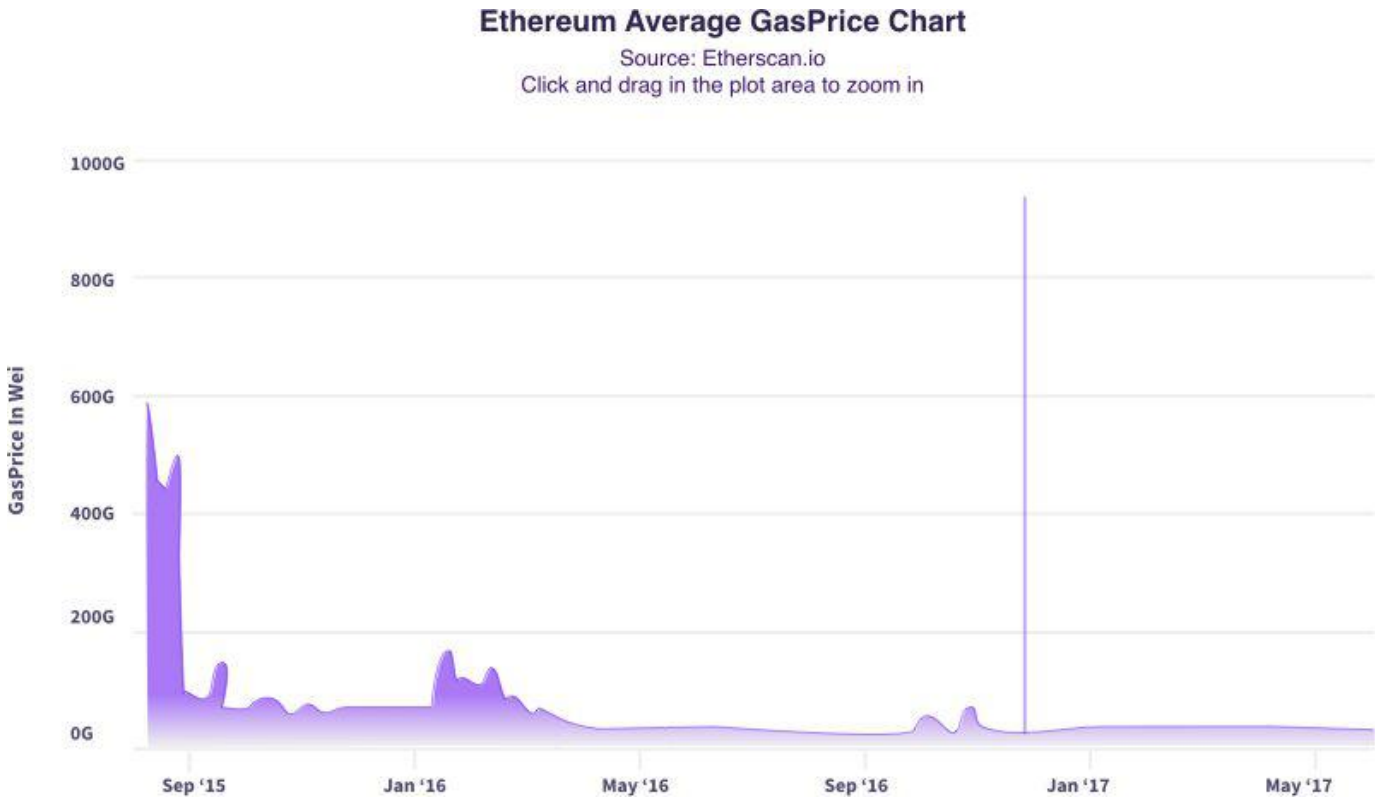
“Gas” is a unit of Ethereum compute capacity. Technical term “gas consumption” is used to indicate how much gas is needed to perform a specific action on the platform. EWT costs 500 units of gas, saving data when using the Ethereum - 100 units of gas. These numbers are integrated in the software.

Finally, “gas price” means how much each gas unit worth in Ethereum.

The total cost of any action on the Ethereum is calculated by multiplying the gas consumption by the gas price. If the gas price remains permanent and the cost of the Ethereum is growing, like it happened recently, the cost of executing smart contracts is enhancing.

Pressure on miners

The gas price in Ethereum is set by miners. Some people expect that they will abate the gas price. Some of this decision supporters espousing point that if the miners do not do this, then fewer people will use the network and pay commission fees. However, the gas price is still constant.



The gas price occasionally hesitates, as shown on the Etherscan chart, but recently it is mainly in the range 22-23 GWei (this is 0.000000000000000022 ETH or less than one cent).

Thus, the current situation can be changed by reducing the commission fees by the miners. Some users have even launched a campaign to force the mining pools lower the gas price.

Considering this, the reasons to worry about the price increase become less. As far as the blocks remain blank and many of them are generally empty, then users do not compete for the transaction to hit the block offering higher commissions. The gas price does not decrease yet the blocks are not filled so everyone loses from this. Essentially, it happens due to the developers' refusal to act. Developers must establish a dynamic gas price, which will depend on the fullness of the blocks in particular.

On that basis, we offer a solution that significantly reduces the gas cost at the platform level and consequently all transaction expenses.

2. YOC keeps up with the times

YOC is not just a new start-up, it is a steadily evolutive and growing platform that was launched in 2016. At the same time growth since September equaled a very significant 300%. Furthermore, the tasks that have set at the construction phase of this platform are mostly fulfilled and it's time to move on in order to become maximum useful for our users.

Switching to the Ethash algorithm

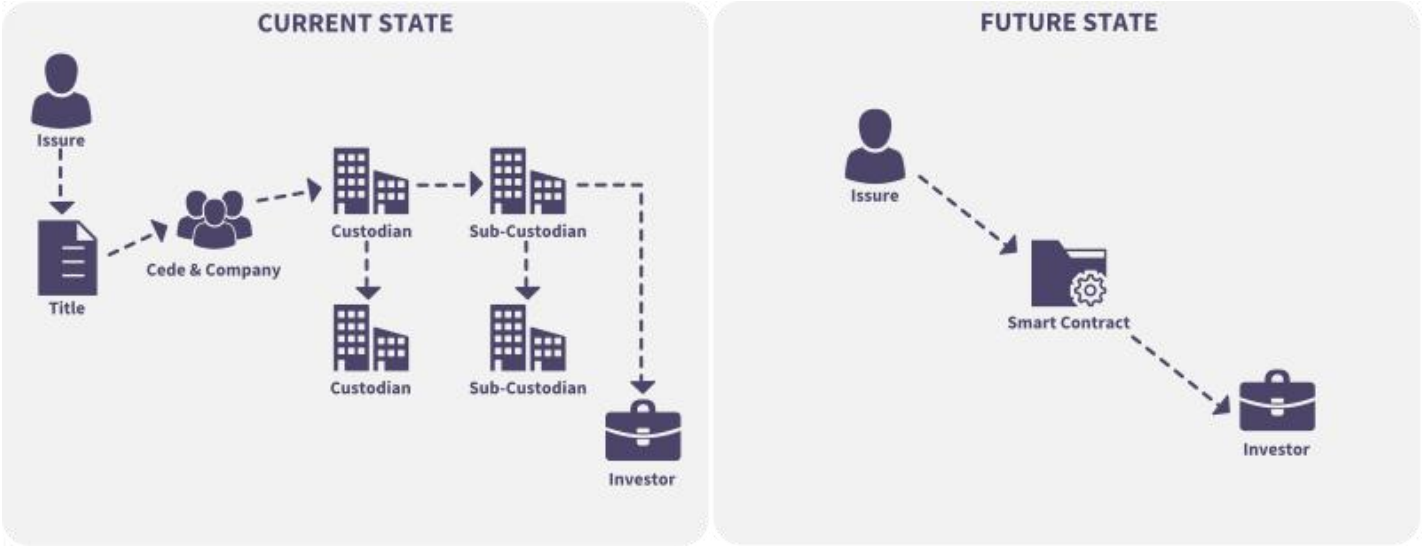
Basically, the Ethash algorithm is an improved Scrypt algorithm on which Yocoin was previously implemented. The platform itself provides undeniable advantages in the use of smart contracts and Dapps.

Advantages of smart contracts:

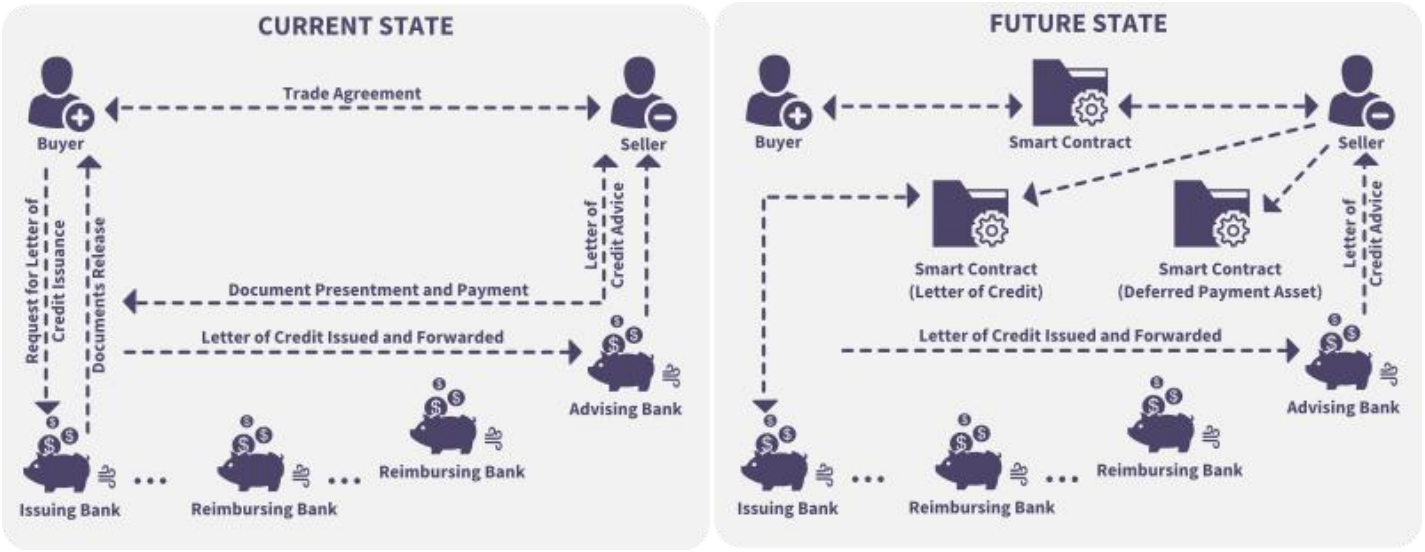
1. Time rate and updates in real time – by automating tasks that are often performed manually smart contracts can significantly speed up workflow.
2. Reliability: besides reducing the risk of mechanical error, the decentralized nature of the Blockchain ensures that data can be recovered if lost. At the same time, cryptographic protection of the platform practically excludes the possibility of hacking.
3. The decentralized nature of the platform eliminates the risk of manipulation.
4. Cost optimization = reduction of intermediaries number + cutting down expenses in m/hrs.
5. Opportunities for new business models: a private network for exchange of solar energy based on Ethereum smart contracts is being created in New York.

The main business dimensions suitable for smart contracts:

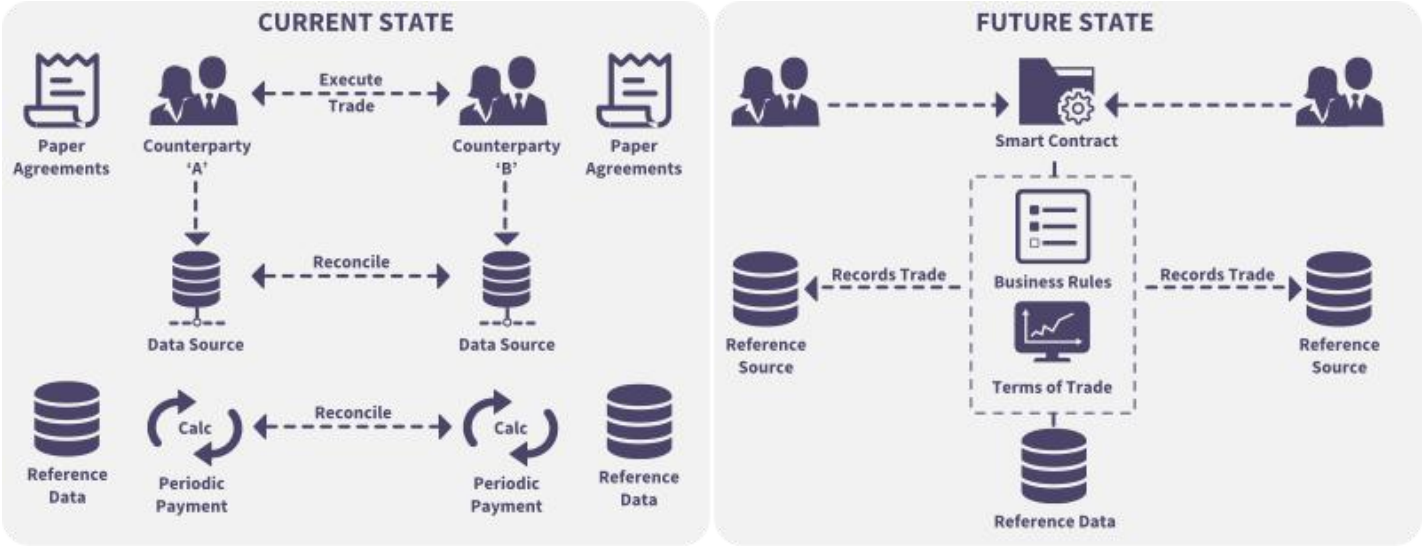
Securities processing: smart contracts will help to automate the dividends payment, additional share issue and the voting process at the global meeting of shareholders, as well as reduce the costs and risks associated with the presence of counterparties and the maintenance of inefficient paper documentation.



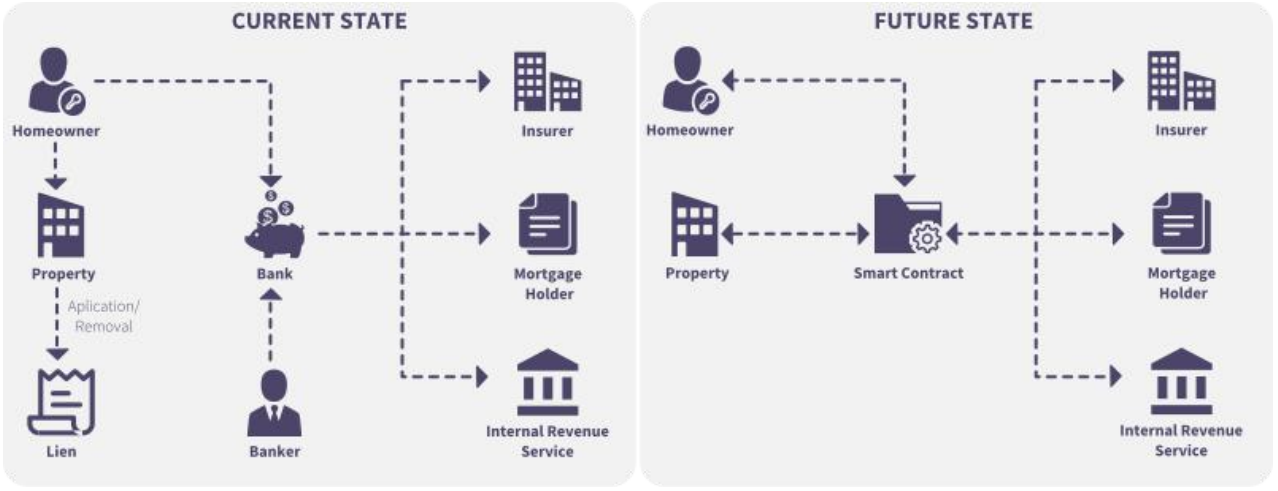
- Trade financing: execution conditions of the letter of credit registered in a smart contract will be checked automatically which will speed up the process and increase the efficiency of trade.



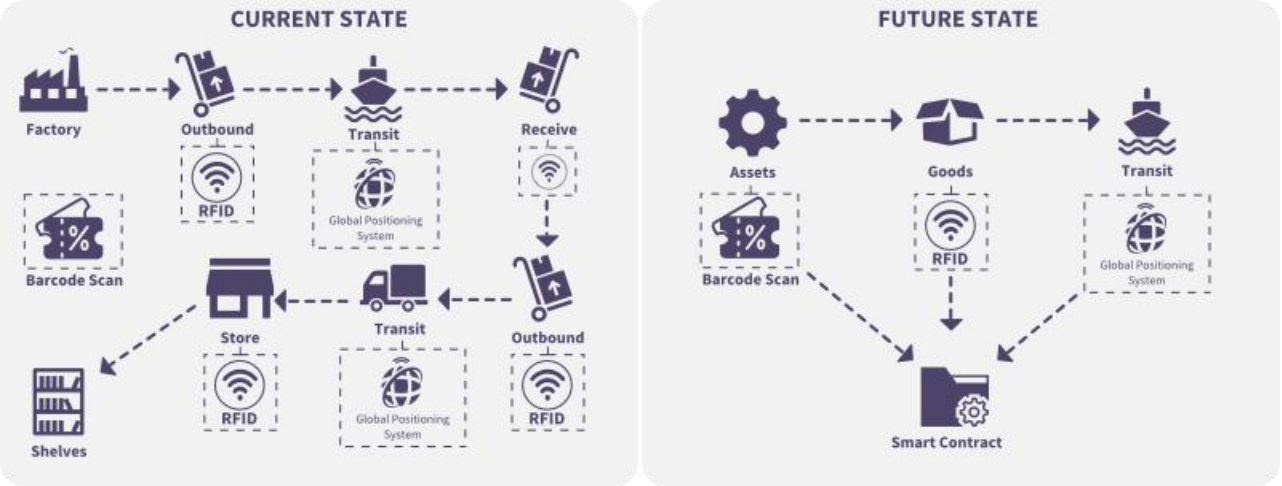
Derivative maintenance: processes of OTC derivatives maintenance are time consuming, as whereas all counterparties are engaged in this independently. Smart contracts can optimize these processes by carrying out regular actions (f.ex. periodical payments), in accordance with information coming from the oracle.



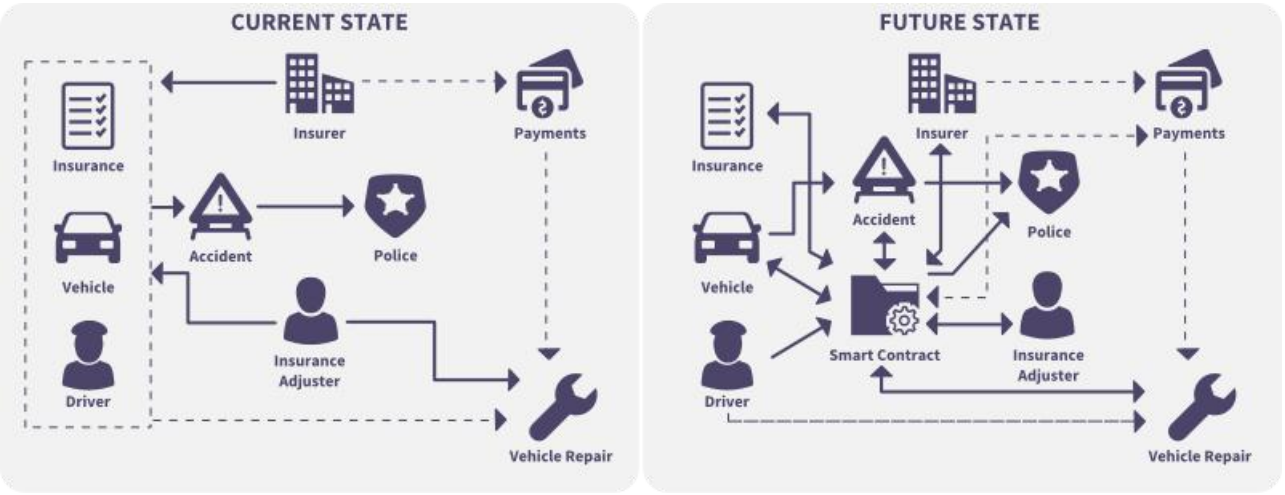
- Mortgage: smart contracts can be used in payments withdrawal and removing a mortgage lien from the collateral after the loan settlement.



- Supply Chain Management: smart contracts will make it easier and more valuable to track deliveries in a single chain linking production, logistics, retail and the counterparties involved in it.



- Vehicle insurance: insurance companies will be able to reduce significantly the cost of collecting and verifying the authenticity of various documents through the implementation of smart contracts. They will contain the terms of hedging and incoming data from various car sensors will prune the calculation of insurance payments.



Other business dimensions are discussed in the table below:

Indusrty	Practice
Public area	Voting - voices are recording in Blockchain, smart contract applies specified criteria to the results and performs certain actions (f.ex. announces the winner)
Health care	Electronic reminder file - smart contract provides access to the patient's file with the consent of the parties - the doctor and the patient must authorize the actions with their electronic keys Health monitoring - analyzing the data that comes from different devices (IOT), smart contract can automatically reward the patient for achieving certain goals
Energy and resources	Automatic charging stations for e-cars - at the beginning of the use the smart contract freezes money and opens access to the charging station. After the end of the charge, the balance is automatically defrosted
Technologies and media	Royalty distribution - according to predetermined shares, a smart contract calculates and makes payments to copyright holders

Apps

The Yocoin platform is also not deprived of the ability to create and deploy Decentralized apps based on it.

Principal categories of these apps are:

The First category - financial apps that provide users with powerful ways to manage contracts and award of contracts for their assets. Examples of such apps – sub currencies, financial derivatives, hedging contracts, savings wallets, wills and even some types of employment contracts.

The Second category – semi financial apps, apps involving also non-monetary tangible benefits; perfect example here is self-executing rewards for solving significant computing tasks.

The Third category – non-financial apps such as online voting and decentralized management.

3. Marketing analysis

Today we are witnessing the development and implementation of completely new technologies that can change the world, as the Internet did once. Blockchain is one of these technologies.

And if, figuratively speaking, the Internet has trimmed off distances between people, the technology of Blockchain is aimed at minimizing mistrust between people. This is dramatically important since the existence of notarial system, registry-based activities, banks, guarantors and many other institutions of modern civilized society is based precisely on the lack of trust between people.

Undoubtedly, this technology will not be able to clear the world from intermediaries but it is already apparent that it will significantly simplify the economic turnover, so that modern business models will become more efficient, and transaction expenses will be extremely reduced.

Thanks to the existence of Blockchain technology difficult-to-implement ideas and conceptions such as internet of things, shareconomy and smart contracts will gain traction.

It is smart contracts and their derivatives Decentralized apps form the basis of some cryptocurrency platforms profit source (f.ex. Ethereum). The more gas is spent on fulfilling the conditions of smart contracts the greater benefit the Ethereum network and the miners have, who have a commission from all transactions that occurred within this block in addition to the reward for finding a block. On the one hand, it's remarkable and makes Ethereum a very attractive coin for both miners and investors, who keep their savings in this coin. On the other hand, for final users of this service, which are smart contracts, it composes a great amount of fixed expenses. Eventually, if smart contract is used as a token for an ICO project, in this case, the organizers of this ICO do not really think about the costs associated with the implementation of the negotiated smart contract. As the main goal of this campaign is to raise investors' funds and these expenses will have little impact on the company's budget in case of successful ICO. Therefore, the difference between 1 and 10 dollars for the execution of this contract does not play any role at all because the ultimate goal is to maximize the funds raising.

Now let’s study another situation. Supposing that smart contract or DAO is a kind of service where customers make payments with each other on an ongoing basis regularly store various data in Blockchain network etc. At the same time, the price of this service for the final user, the demand of which arises quite often, will play a significant role in the success of this service.

Let's try to illustrate this in the context of a decentralized poker room. In this process, players are interested in that the very first particular hand will actually be generated by real random number generator, whereof information will be transferred to the Blockchain network. And as we know, any scope of information transferred to this network comes at a price. Thereunto data transmission is required at maximum speed. At the same time, there is no direct commercial interest in the transfer of this information and a fairly large number of these hands are formed during the game therefore the sum of usage of the network by the Blockchain can be very significant. Normally the poker room charges a certain commission (rake) from each dealing and in order to be competitive in this market it will have to cover these expenses by reducing its own commission or by charging commission from the players. In either of these cases, the profitability of this intention is decreasing. Taking into account that this operation will take place in Ethereum Blockchain and

consider its value of 1000 USD, the calculation shows that saving data about one dealing will cost approximately 0.18 USD. At this cost of operation, this poker room will be able to provide a game with limits of at least 1-2 USD Texas Holdem and tournaments with a buy-in of at least 100 USD. But there are few players like we defined. On the assumption of this logic, any decentralized poker room is interested in retaining this commission as low as possible. Moreover, in the logic of the poker room there are a few really valuable transactions it's basically just the operations of inputting funds when replenishing and withdrawing money from your account.

Exactly for such purposes, Yocoin platform was created for; as a comparison, the same commission will be 0.000036USD. In this case, the poker room will be able to provide a game with a limit of 0.01-0.02 USD Texas Holdem. Which are traditionally minimal for any of the existing ones.

In addition, there is striking example of how a large operating commission can negatively affect the use of Blockchain network. Now, let us look back at STEAM's refusal to accept BTC (Bitcoin) as a form of payment after the commission amount was over 20 USD. Although it was a sufficiently large player who popularized this cryptocurrency indeed and provided it with viable high-demand and quality goods in a large volume.

These examples show that workable Blockchain business processes require a platform with minimal commissions. As you correctly understand this platform is YOC.

4. Technical part & platform components

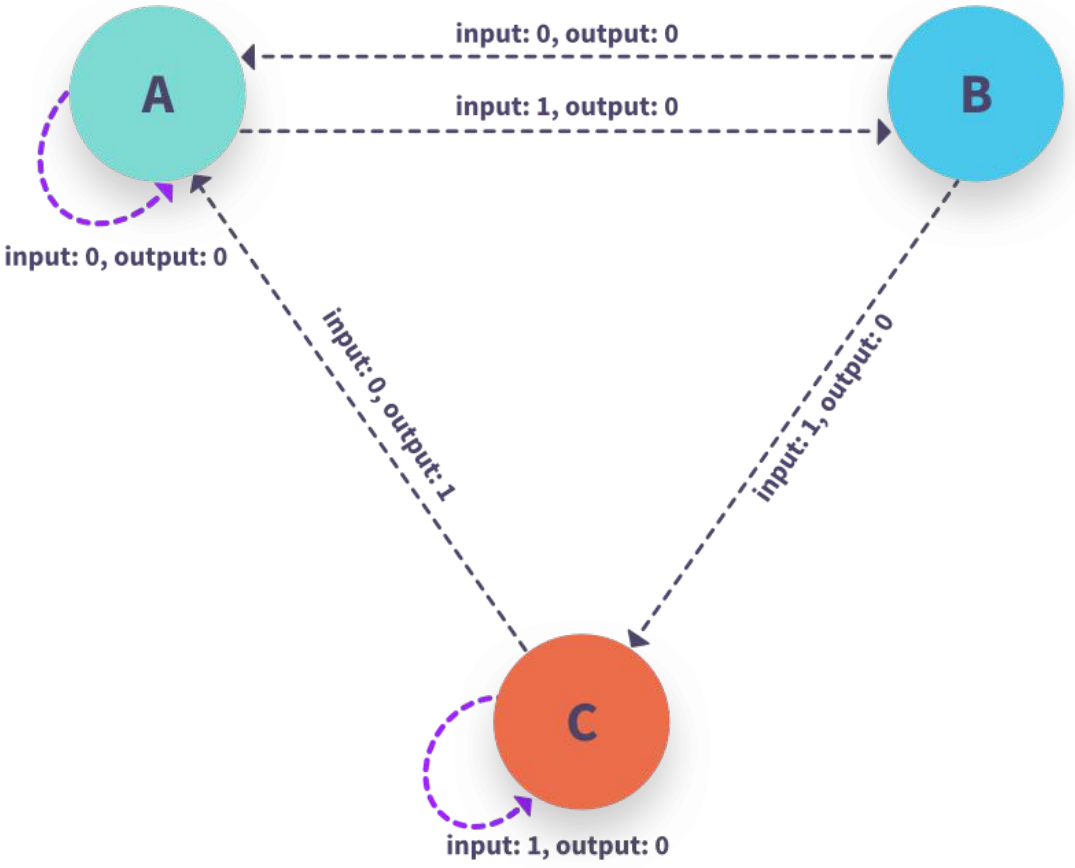
What offers Yocoin platform and how it works?

Yocoin is a full-fledged platform for guaranteed processing. The main advantages can be stated as follows:

- user authentication through cryptographic signatures;
- fully customizable transaction logic and state changes;
- is resistant to DDoS attacks;
- no single point of network failure;
- history of all network activities is stored in the public domain in decentralized distributed database (Blockchain).

Yocoin platform Blockchain paradigm.

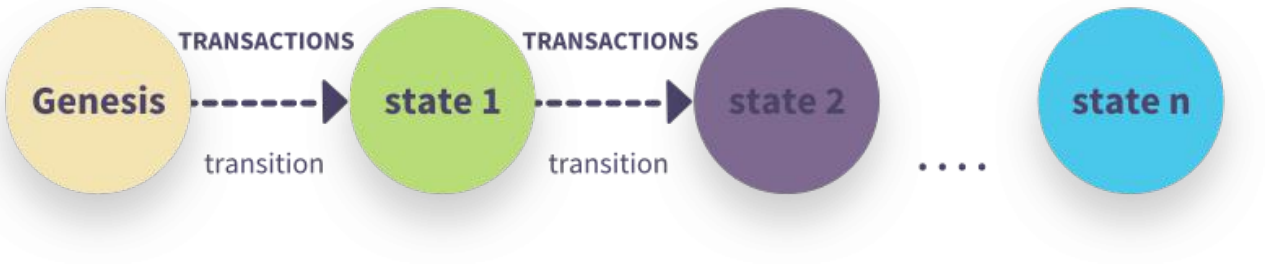
In fact, Yocoin Blockchain is a transaction state system. IT define terms “state system” or “state machine” as a system that processes input information and on its basis transform into a new state.



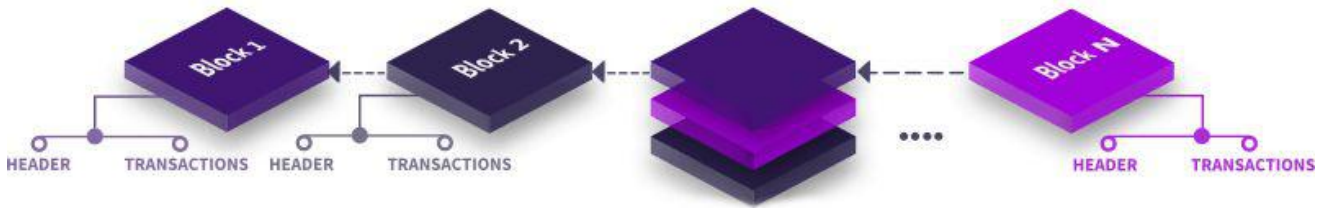
In Yocoin state machine, all processes start with an “genesis state”.

This state is an analog of the machine null state until the moment when any transactions related activities begin to occur in its network.

When the actions begin to occur the genesis state is replaced by the final state and at any time, the final state displays the current state of Yocoin.



The Yocoin state has millions of transactions. These transactions are arranged into ”blocks”. The block contains a number of transactions herewith each subsequent block is connected to the previous one so a unique chain of blocks is provided.



Transaction must be adequate in order to commence its transition from one state to another. The transaction is considered adequate only when it has passed the verification process - so-called “mining”.

Mining - when a group of nodes (computers) spend their computing resources to create a block of correct transactions.

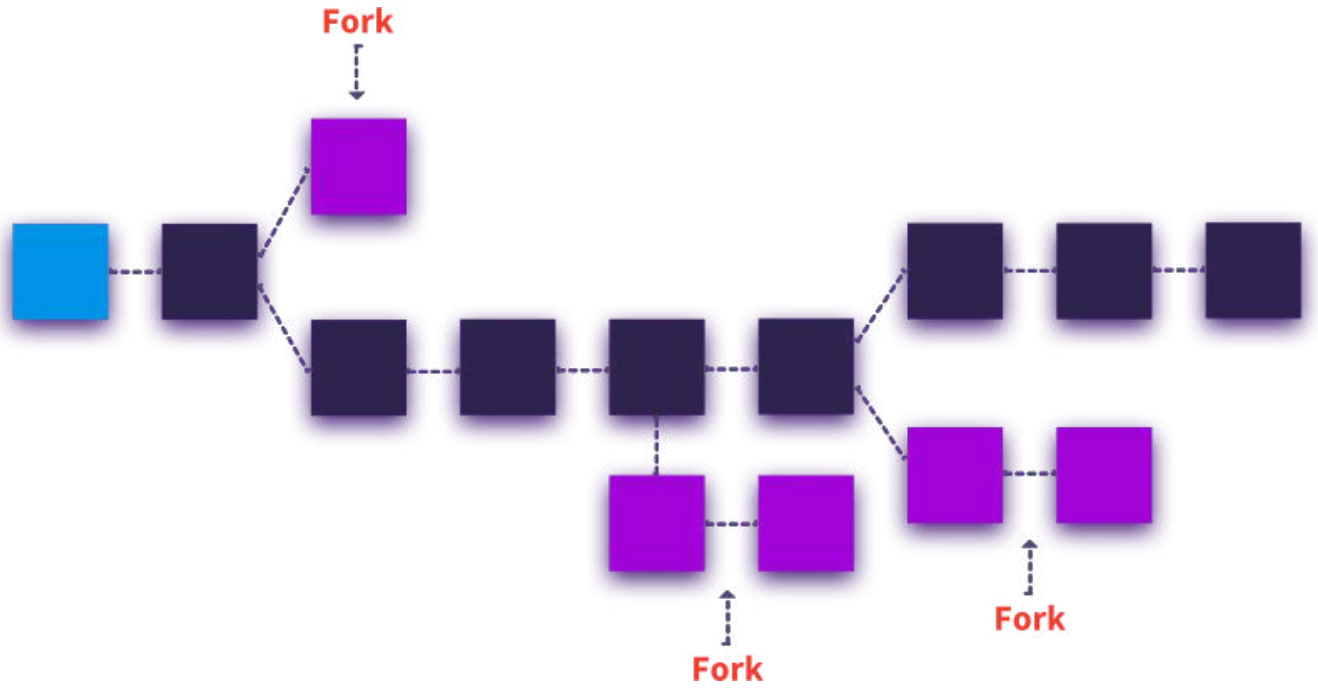
Any network node that claims to be a miner can attempt to create and test a transaction block. The attempts of dozens of miners to simultaneously create and verify a block of transactions is a common experience. Every miner provides his mathematical «proof» when sending a block to the Blockchain, which acts as a kind of guarantee: the transactions in the block are considered correct in case existing of the proof.

Miner must provide his mathematical proof faster than any other competitor does in order to add his block to the main Blockchain. The process of checking each block, which consists in providing the miner with its mathematical proof, is called “proof of work”.

Miner who validates the new block receives a certain reward for doing this work. What kind of reward is it? Yocoin Blockchain uses the built-in digital token, which is called “yoc”. Each and every time a miner validates his block of transactions a new token or a new yoc is created and the miner gets reward for creating it. You can have quite a logical question: is there a guarantee that each miner will adhere to only one chain of blocks? How can I be sure that another team of miners will not decide to create their own chain of blocks?

To clarify this, it is necessary to use the term ”transactional one-element system with a global state” which is essentially a Blockchain. Based on this definition, we can conclude that there are no two or more correct current states - it is the only one of its kind. Thus, anyone who takes part in the process of new blocks validating must accept this assertion as truth. The presence of several states (or chains) would destroy the entire system, because it would be impossible to agree on which of the states is correct. For example, imagine that there are several chains of blocks. Then in theory, you can collect 10 coins on the first chain, on the second - 20 coins, on the third - 40 coins, etc. In such circumstances, it is impossible to determine which chain is the most “correct”.

Whenever multiple paths are generated, a ”forking” occurs. Frequently, forkings are very undesirable because they violate the integrity of the system and users have to choose one of the possible chains.



To determine which of the possible paths is correct, and to prevent the formation of multiple chains, Ethereum uses a method called the “GHOST protocol”.

GHOST – Greedy Heaviest Observed Subtree

The GHOST protocol declares that we must select only the path, which performed the largest number of calculations. To determine such path, you can use the number of the last block (“leaf block”). Thanks to this approach, it is possible to determine the total number of blocks in the current path (without taking into account the genesis state block). The higher the block is, the longer the path and the more validations miners must provide. Based on these reasons only correct version for the current state is deemed.

Now let's gasp the main components of the Yocoin system:

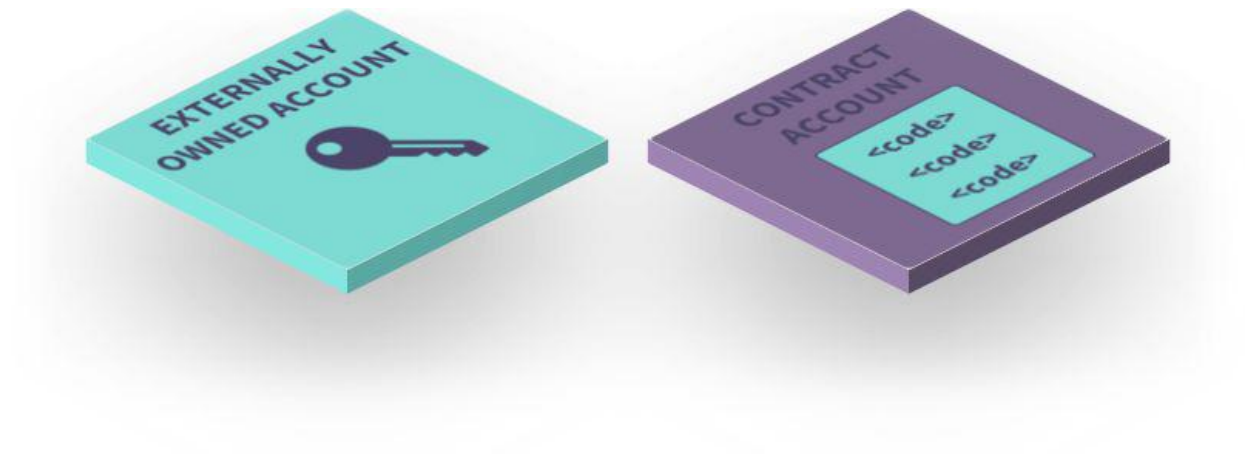
- accounts
- state
- gas
- transactions
- blocks
- transactions flow
- mining
- validation

Accounts

The global state of the Yocoin platform consists of many small objects –accounts that interact with each other due to the messaging paradigm. Each account has a specific state and a 20-byte address. Yocoin address is the 160-bit identifier used to trace any of the user accounts.

There are two types of user accounts:

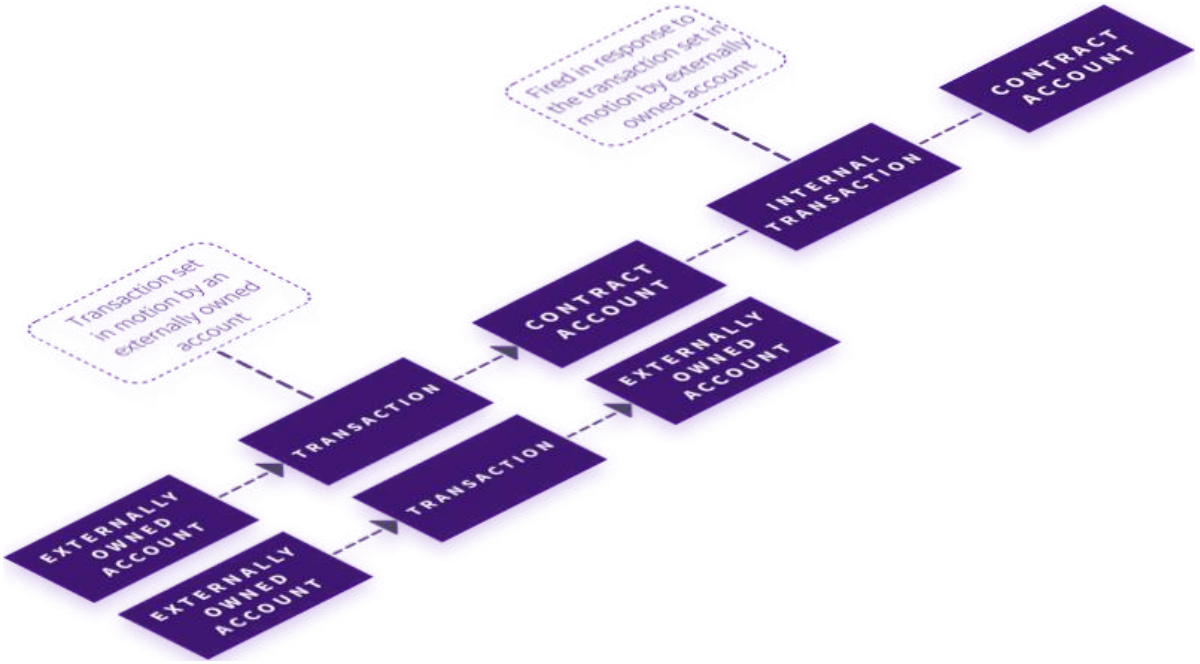
- externally owned accounts are controlled by privacy keys. However, such records do not have any code related to them.
- contract accounts are controlled by a special code specified in the terms of the contract and have a code related to them.



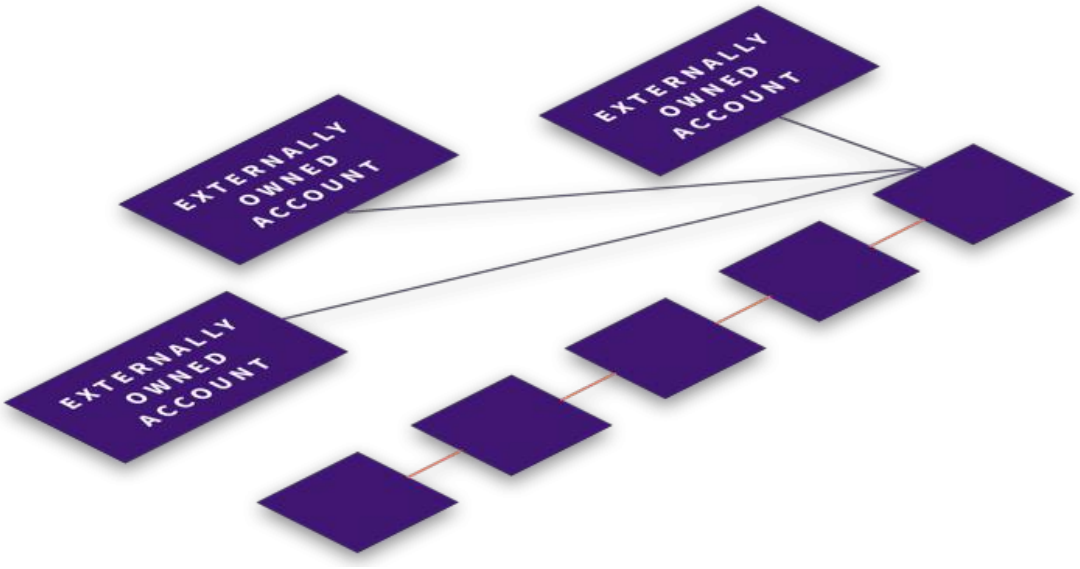
Externally owned and contract accounts:

Let’s clarify the main differences between externally owned and contract accounts. For an externally owned account, it is possible to send messages to other externally owned accounts as well as to contract accounts. For this purpose, you must create and register a new transaction using the privacy key. The message between the two externally owned accounts is just the value for the transfer. On the other hand, a message sent from an externally owned account to a contract account involves activating the contract account code. This allows you to perform certain actions (f.ex. you can transfer tokens, record values to the built-in memory, create tokens, perform certain calculations, create new contracts, etc.).

By dint of contract accounts, it is impossible to initiate new transactions independently rather than with externally owned accounts. Instead of it, you can only run transactions in response to other received transactions when operating contract user accounts (f.ex. received from an externally owned account or from another contract account). More information about calls between contract accounts you can find in paragraph “Transactions and messages”.



Each action in the Yocoin Blockchain occurs through transactions initiated by externally owned accounts.



Yocoin Blockchain

Accounts states

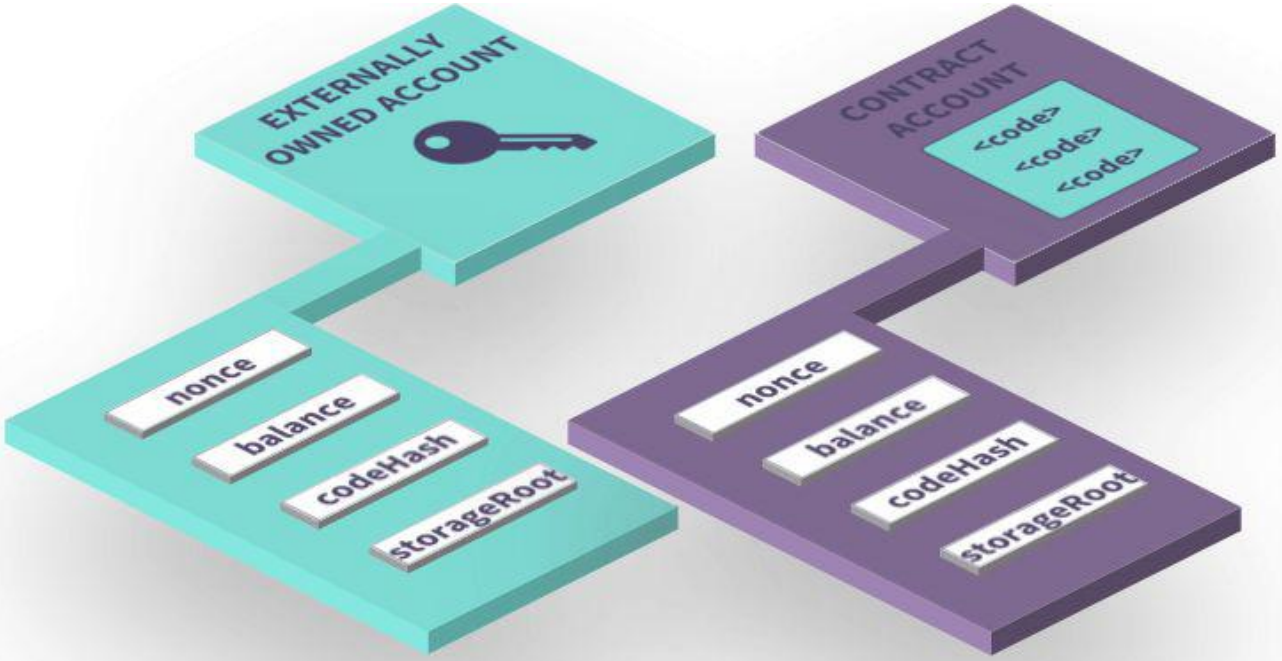
The state of each account, regardless of their type, can take one of four values:

nonce: if the current account corresponds to an externally owned account, the number obtained is the number of transactions that have been sent from the account address. If the account is a contract account, the nonce element is the number of contracts created though this account.

balance: sum of Wei purchased by this account. For example, each yoc, which is the exchange unit of Yocoin, contains 10 ^18 Wei - fractional parts of yoc.

storageRoot: root node hash of the Merkle Patricia trie. The Merkle trie encodes the contents hash of this account and is blank by default.

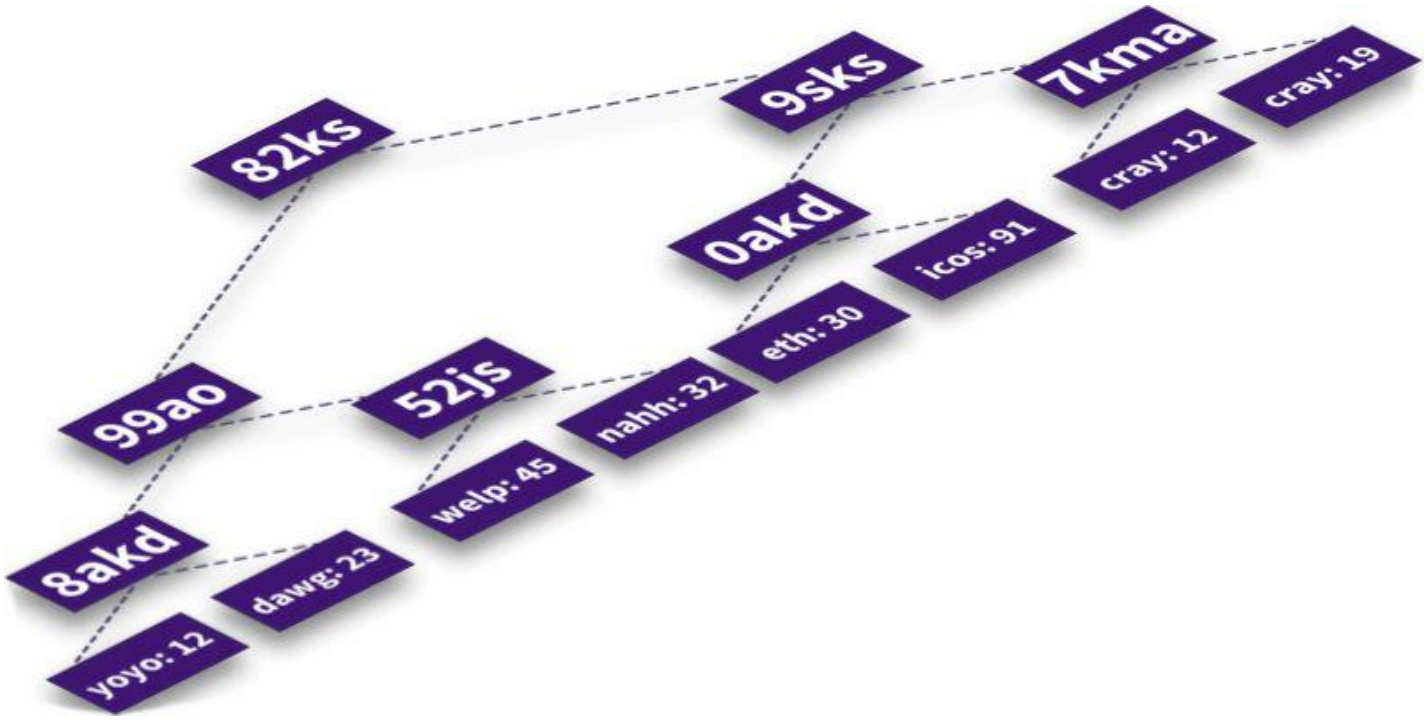
codeHash: hash of the account's YVM-code. This field is a code that is hashed and stored as codeHash for contract accounts.



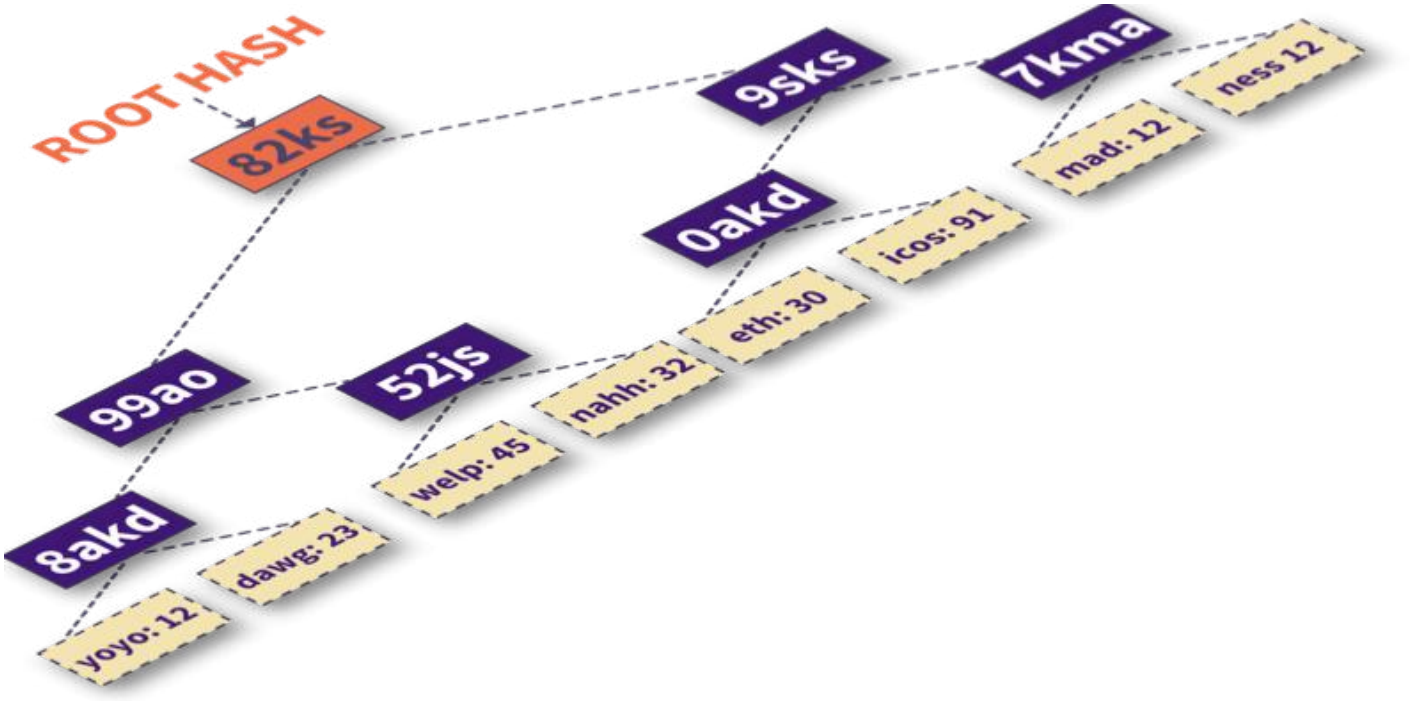
Global system state

So, we figured out that Yocoin's global state is a mapping between account addresses and account states. This mapping is stored in the data structure - Merkle Patricia trie. Merkle trie is a type of binary file consisting of a set of nodes that includes:

- certain number of leaf nodes that are located at the bottom of the trie containing the base data
- set of intermediate nodes, with each node representing a hash of its two child nodes
- one root node, also formed from the hash of two child nodes, which represents the top of the trie

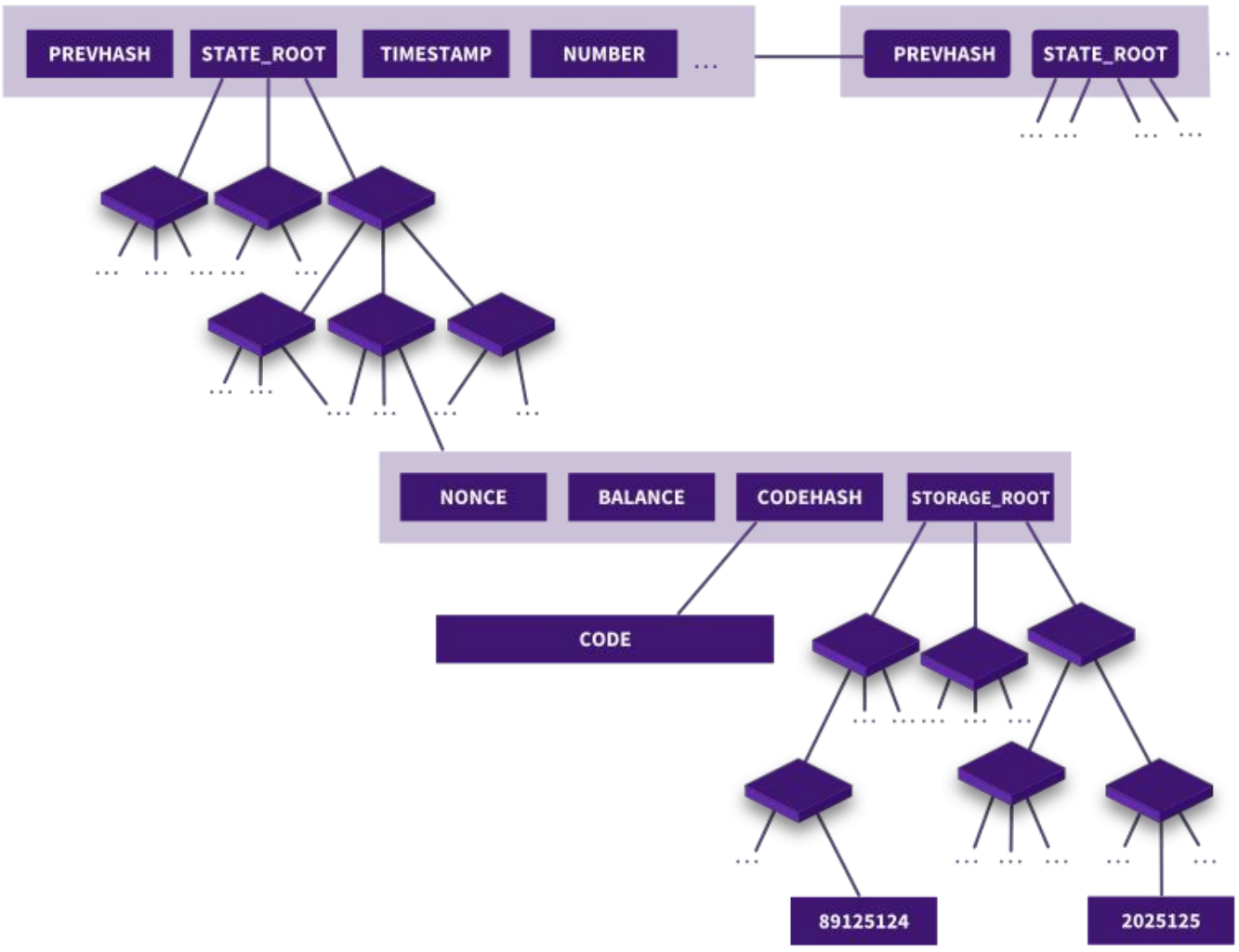


The data at the bottom of the trie is created by dividing the data that we want to save into separate fragments. Then these fragments are placed in the data storage baskets, after which their hashing is performed and a similar process is repeated until the total number of hashes is equal to unity root hash.



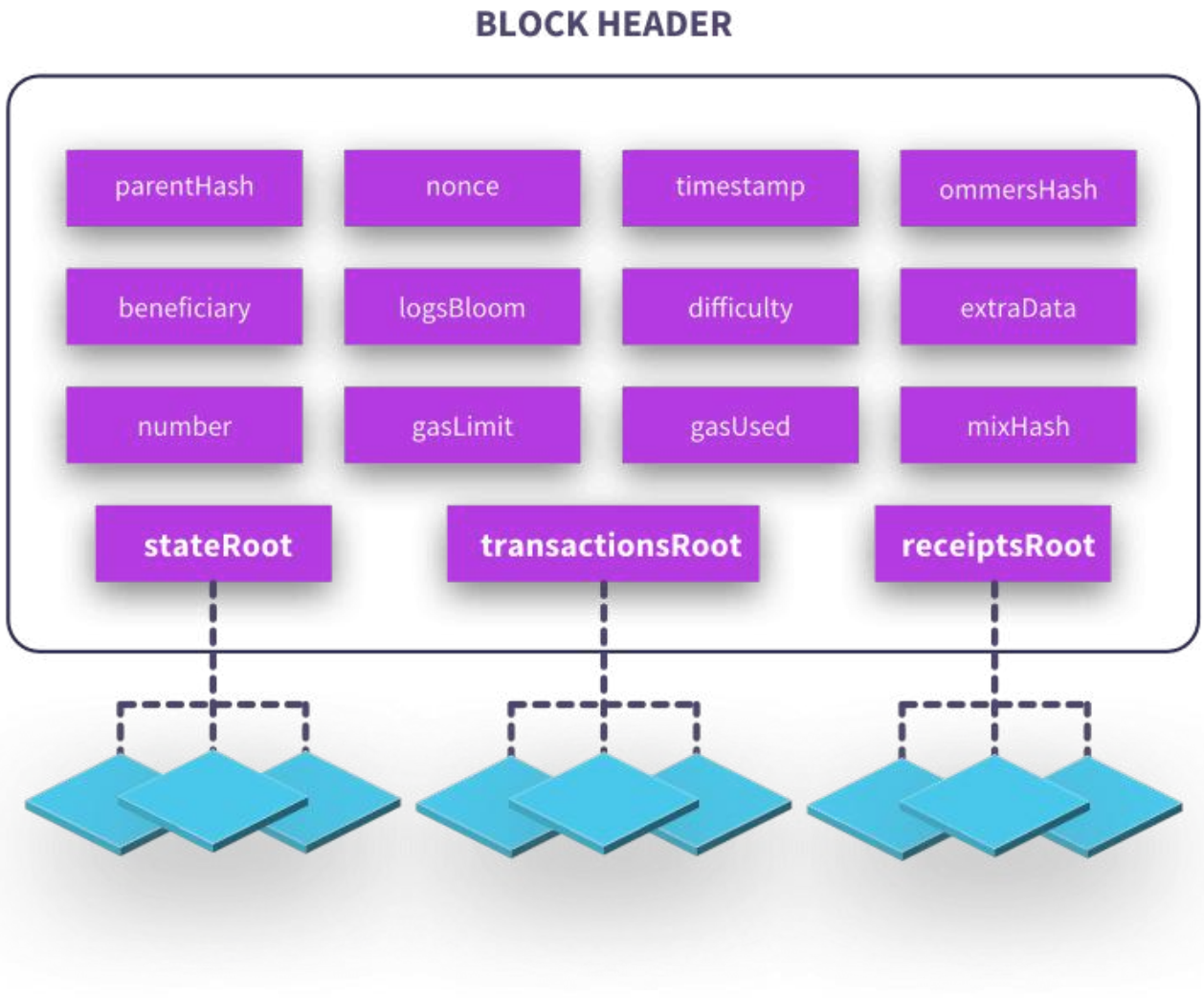
For each value stored inside this trie, you will need to enter a specific key. To obtain the relevant value stored in the leaf nodes you should get the key command: he chain of which child node must be followed. As for Yocoin, the display of the key/value required for the state trie is

between the addresses and associated accounts, including balance, nonce, codeHash, and storageRoot for each of the accounts, while the storageRoot is being a trie.



A similar trie structure can also be used to store both transactions and payment receipt pages. Considering this in more detail, each block has a so-called “header” or header file where is stored the root node hash of the three different Merkle trie structures, including:

- trie state
- trie transactions
- trie payment acceptance pages

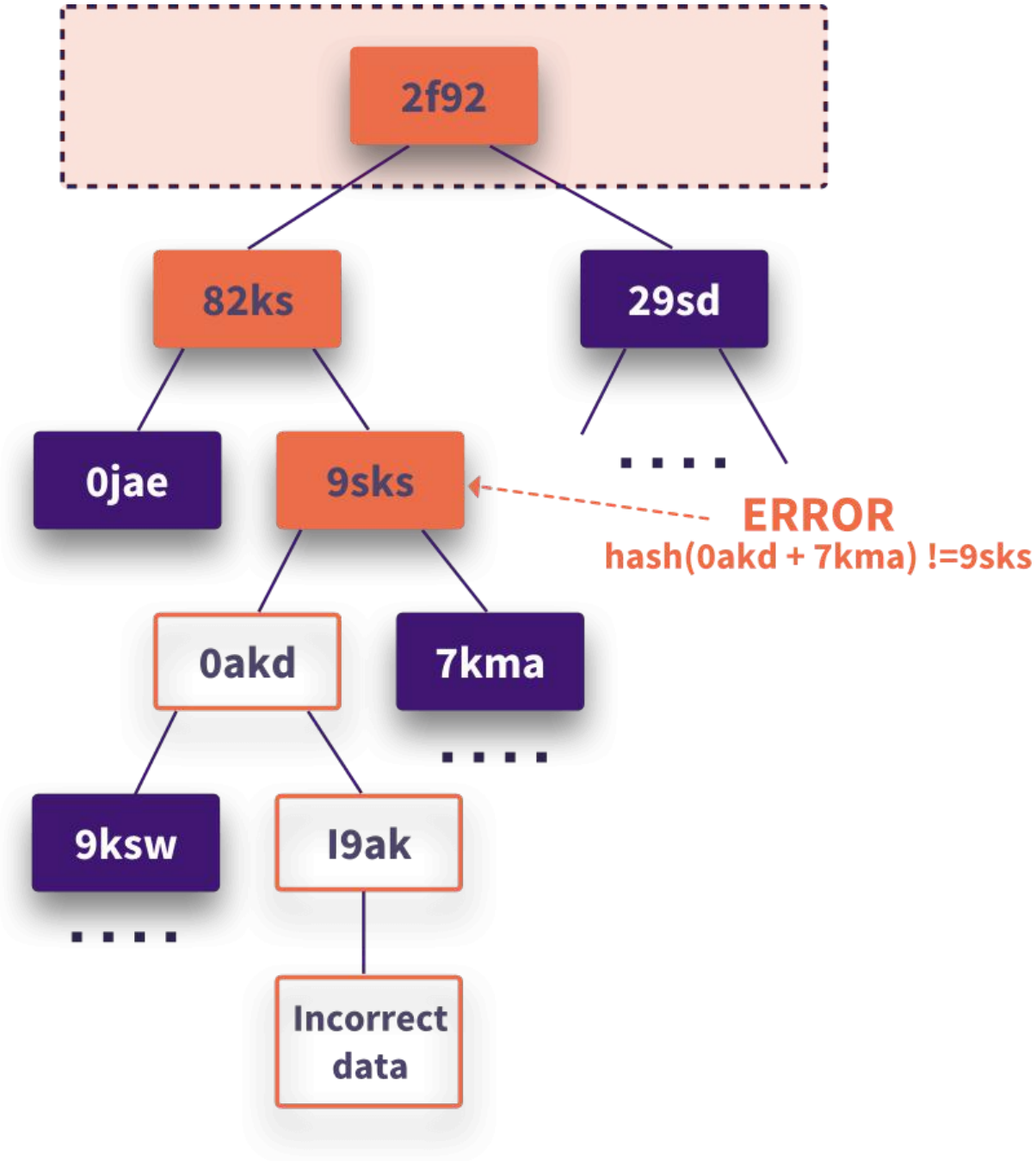


The ability to store this information effectively in Yocoin Merkle trie is an incredibly practical solution for so-called lightweight clients or lightweight nodes. It worth mentioning, that Blockchain support is provided with a set of nodes. In lay terms: there are two types of nodes, full and lightweight.

The full archive node synchronizes Blockchain by loading the entire chain from the genesis state block to the current block containing the header file herewith all transactions in it are executed. Generally, miners store the entire archive node, since without it, they will not have the opportunity to participate in the mining process. In addition, you can also load a full node, and there is no need to execute each individual transaction. It is also stands to mention that each full node always contains a full chain.

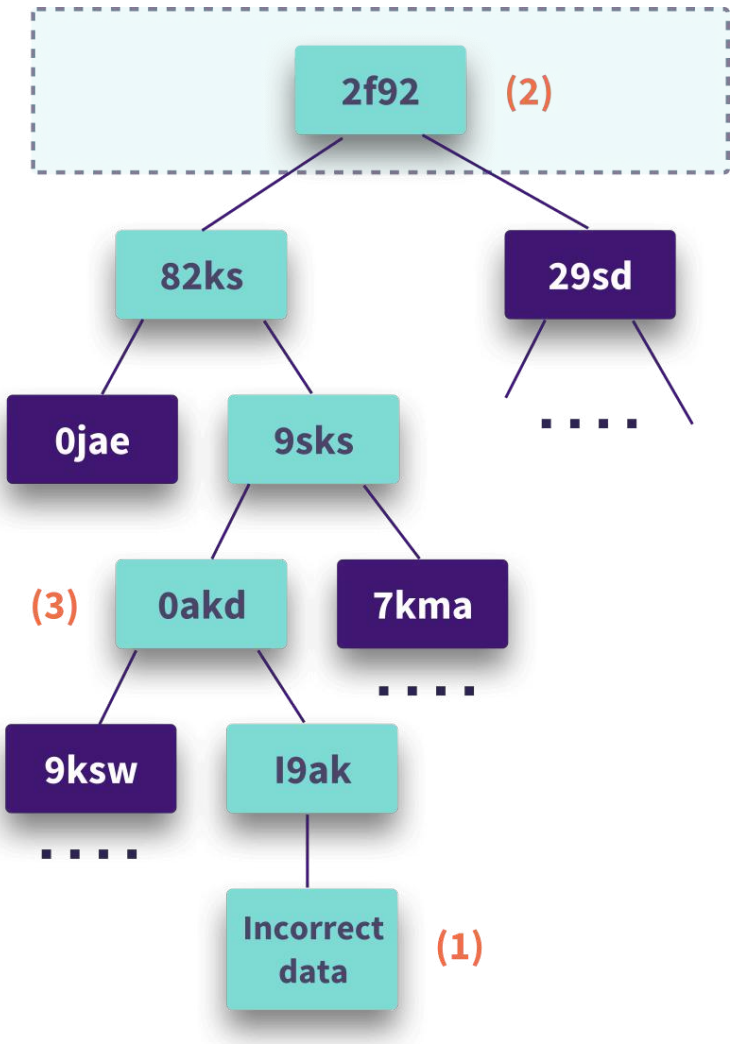
As long as it is not necessary for a node to execute each separate transaction or to request the accumulated data, storing the full chain can be superfluous. Exactly in this case we come across such concept as a lightweight node. Instead of loading and storing the complete chain, as well as performing all transactions, lightweight nodes load only a chain of header files from the genesis state block to the current header, and no transactions are performed. Whereas lightweight nodes have access to block headers containing a hash of three tries, they can easily create and receive appropriate responses regarding transactions, events, balance, etc.

The hash in the Merkle trie expands from the lower branches to the upper ones, and if an attacker tries to replace the original transaction with a counterfeit at the bottom of the Merkle trie, this will change the hash of the top node, which in turn will change the hash of the node above it and so long as, ultimately, it will cause the root changing.



Any node that requires testing any part of data uses the so-called “Merkle proof”. “Merkle proof” consists of:

- fragment of data to be checked
- trie root hash
- “forking”- all hashes, from the checked data fragment to the root



Each user who perceives this proof can check whether the hash for a particular forking is appropriate for the entire section of the trie, and whether the fragment takes the corresponding position in that trie.

As can be seen from the above, we can elict that the advantage of using the Merkle trie is that the root node of this structure is cryptographically dependent on the data stored in the trie. Thereby, the hash of the root node can be used as a secure identifier for this data. Since the header of the Blockchain includes the root hash of tries, as well as their states, transactions and payment arrival information, any of the nodes can check this or that part of the Yocoin state without having to store all states that can potentially be unlimited in size.

Gas and rewards

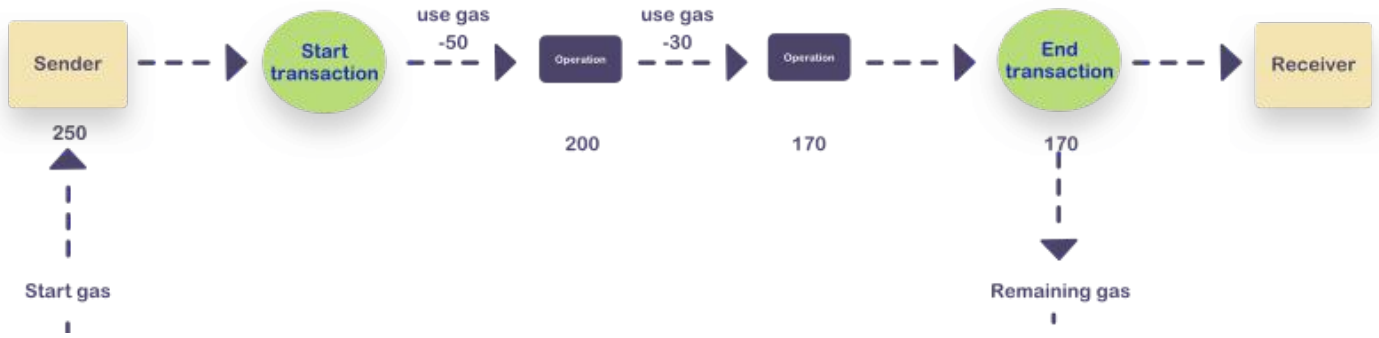
Payment process is one of the essential points in Ethereum System. For any computations performed as a result of transactions within the Ethereum network, a certain fee is taken. The nominal of this payment is called “gas”.

Gas - is a measurement unit that is used to determine the amount of reward for a specific computing. Gas price - is the amount of “yoc” you can spend on each gas unit. Gas price is measured in “GWei”. Wei is the smallest unit of yoc, where 1018 Wei is only 1 yoc. One GWei is equal to 1,000,000,000 Wei.

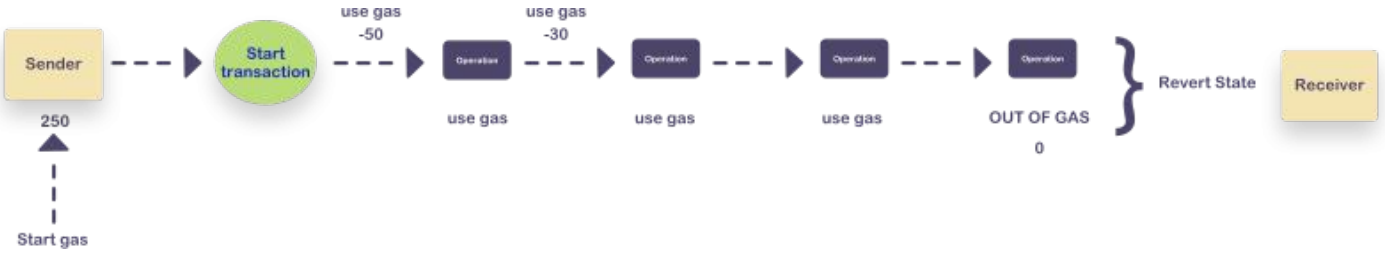
For any transaction, the sender must set a gas limit, as well as gas price. The gas price as well as gas limit is the maximum amount of Wei, which the sender is willing to pay for the transaction.

Let's imagine the sender sets a gas limit of 50,000 GWei, and the gas price is set at 20 GWei. It means that the sender is ready to spend up to 50,000x20 GWei = 1,000,000,000,000,000 Wei or 0,001 yoc for this transaction.

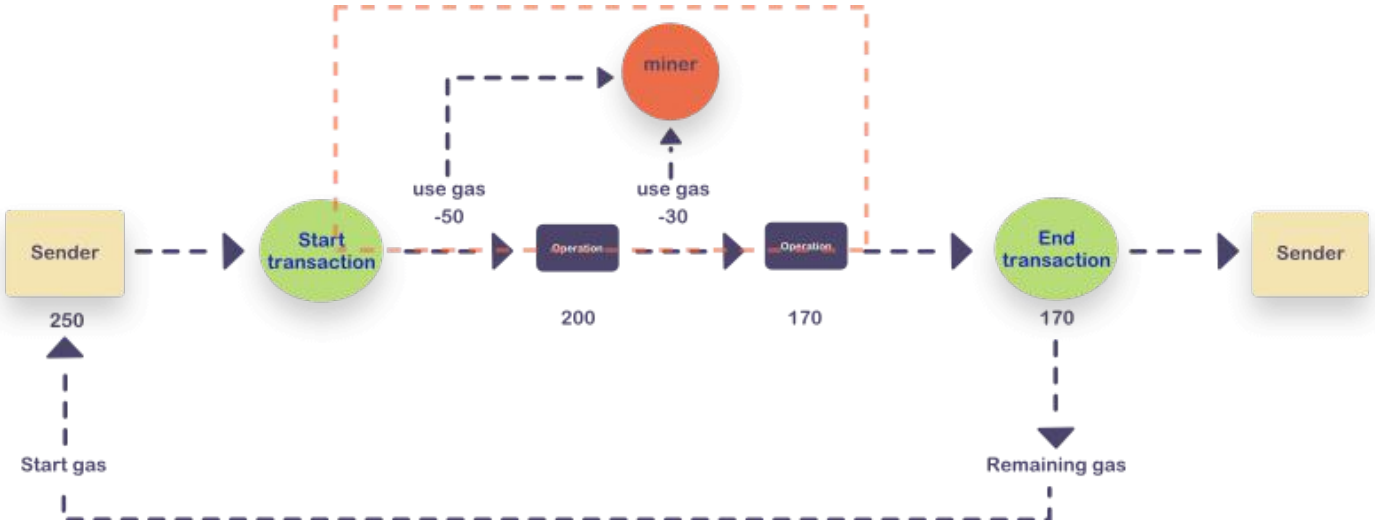
By so doing, gas limit – is max amount of gas that the sender is willing to pay. The sender can execute transactions as long as balance of his account there is enough aether to cover this maximum. Furthermore, the sender will be compensated for any losses associated with incomplete use of the gas at the end of the transaction, with the gas being exchanged at the original rate.



In case when the sender has not sent the required amount of gas for the transaction, the transaction will be carried out "without gas" and will be considered invalid. That is why the transaction is terminated and any state changes are canceled, so the Yocoin system returns the transaction participants to their genesis state. It bears noting, that the information about failed transaction is recorded into the system so that you can track which transactions were performed and at what stage the failure occurred. And what is also critically: before the time the sender has run out of gas, a certain amount of effort has already been spent on the machine to perform the computing, it would be logical to assume that the losses associated with the gas consumption will no longer be reimbursed to the sender.



Where exactly we send the gas? Thus, all the money that was spent on the purchase by the gas sender is sent to the beneficiary address, which in most cases is the miner address. Because miners perform computations and check transaction, they are actually receiving a gas fee as a reward.



Generally, the higher the gas price is, which the sender wants to pay, the higher the reward received by the miner as a result of the transaction. Moreover, the more likely the miner will make a choice in its favor. Thus, the miners are free to choose which transactions they want to validate, and which transactions they can ignore. Frequently miners tell the senders what gas price they should request, so that the first ones are ready to execute the transactions.

Payments for storage using

The gas is used not only to pay for certain computations but also to disburse the storage using. The total storage usage fee is 32 bytes. The issue of storage payment has some nuances. For example, since the augmentation of the storage space involves increasing the size of the Yocoin status database, and this applies to all nodes, you just have the incentive to store only a relatively small amount of data. As can be seen from above, if any of the transaction stages involves the record withdrawal in the storage, then the payment for the performance of this operation is not collected, and due to the release of the place in the storage, the losses will also be reimbursed.

Why charge is required?

An important aspect of Yocoin's work is that any operation that is performed by the network is also performed at the same time by each full node. However, all the steps involved in Yocoin virtual machine computing are extremely expensive. So, Yocoin smart contracts can be quite useful for solving effortless tasks (f.ex., launching simple business logic, verifying signatures, as well as other operations related to crypto currency), as distinct from cases where other, more complex tasks are required: storage files or e-mail, as well as performing tasks from the field of machine learning, which can cause excessive network load. Charge implementation prevents users from taking actions to overload the network.

Yocoin uses Turing complete language. Briefly, Turing machine is a machine that simulates any computer algorithm. Thanks to such a feature, it becomes possible in Yocoin to use loops, and this makes it susceptible to the stopping problem - a problem in the case of which you cannot determine whether the program will function indefinitely or not.

For example, in case when Yocoin would not have a charging system, the attackers could try to disrupt the network by executing an infinite loop inside the transaction, without incurring any losses. Thus, the charging system was implemented precisely to prevent the intentional attacks.

It is likely that you will think: “What do I have to do with it? Why should I pay for the storage?” Well, the entire Yocoin network takes charge both for computing and for using the storage.

Transactions and messages

It has already been written that Yocoin is a transaction state system. In other quarters due to transactions that occur between different accounts, the global state of Yocoin changes or moves from one state to another. The transaction is a cryptographically signed part of an instruction that is first defined by an externally owned account, and then it is ordered and transferred to the Blockchain.

There are two types of transactions: sending messages and contract creating (in other words, such transactions create new contracts in the Yocoin network).

Regardless of type, all transactions contain the following elements:

nonce - the number of transactions that were sent by the sender

gasPrice - the amount of Wei that the sender is ready to pay for the gas unit necessary to complete the transaction

gasLimit - the max amount of gas that the sender is willing to pay for carrying out this transaction. This amount is set and paid in advance before any computations are made.

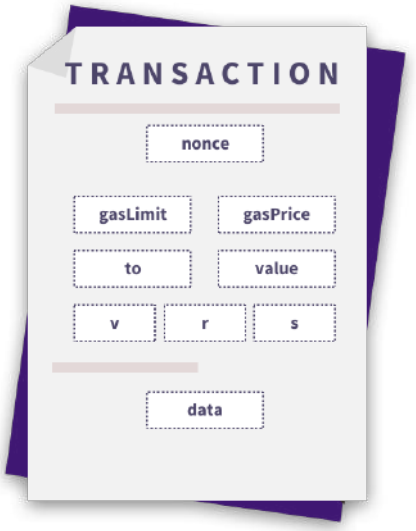
to - address of the recipient. At the time of the transaction associated with the creation of the contract, the address of the contract account does not yet exist, so an empty value is used instead

value - the amount of Wei that will be transferred from the sender to the recipient. In transactions involving the creation of contracts, this value is the starting balance for the newly created account

v, r, s - the designations used to create the signature that identifies the sender of the transaction

init - intended only for transactions involving the creation of contracts. Init is started only once and is not used further. When init is run for the first time, this element returns the body of the account code, which is part of the code that is permanently associated with the contract account

data - the input data (parameters) for calling the message (data is an optional element that is only for message calls). F.ex., if smart contract is a domain registration service, then calling this contract can expect an input field (for example, a domain and an IP address).

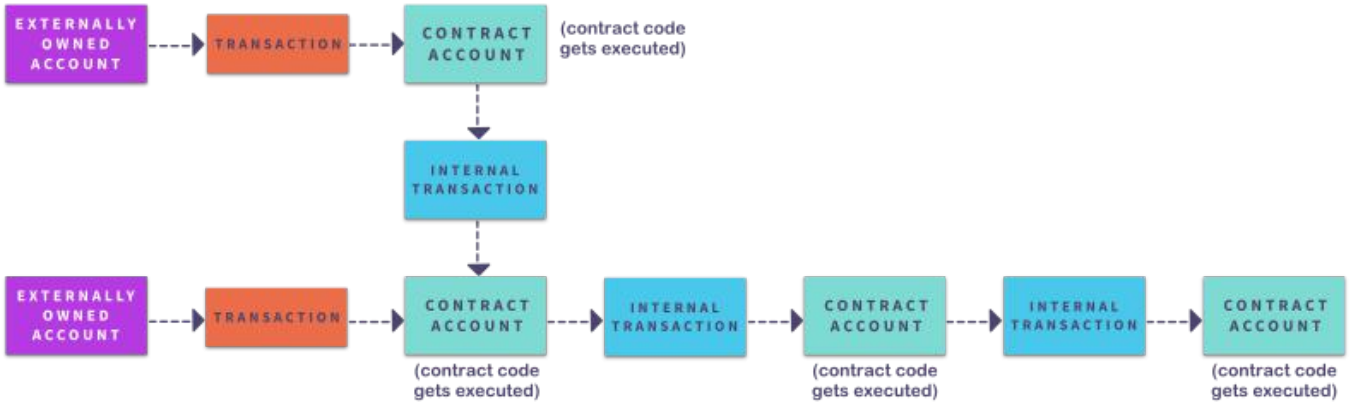


In the “Accounts” section, we found out that the transactions - both for message calls and for creating contracts - are initiated by external accounts, and then redirected to the Blockchain. Otherwise stated, transactions are a kind of bridge connecting the external world and the internal state of the Yocoin platform.

But, this does not mean that one contract cannot interact with another: contracts that are in the global context of the Yocoin state can interact with each other within this context. Their interaction or communication occurs by sending messages or internal transactions. The only difference between internal transactions and the usual ones is that the first ones are not created by

externally owned accounts, but as a result of creating contracts. They are virtual objects, which, unlike transactions, are not aligned and can exist only in the Yocoin runtime.

When one of the contracts sends an internal transaction to another contract, the existing in the recipient's account code is performed.



It is also worth noting, that gasLimit is not provided for internal transactions or messages, since the gas limit is set by the initiator of the original transaction (f.ex., in any account). The gas limit specified by the external account must be high enough to carry out the transaction, including any additional actions that are performed as a result of the transaction, f.ex., the transfer of a message from one contract to another.

In case if in the chain of transactions and messages there is not enough gas to perform one of the last-mentioned, then its execution, as well as the execution of all subsequent messages caused by the genesis execution, will be returned.

Blocks

Anyhow, all transactions are grouped into “blocks”. The Blockchain contains several blocks connected to each other and consist of:

- block headers
- information about the series of transactions included in this block
- series of other block headers for current uncles

What are “Uncles”?

Let's figure out what an uncle is. Uncle is a block whose parent is the mother of the current block. In the first place their presence is justified by the fact that the blocking time in Yocoin is much lower (about 15 seconds) than for other Blockchain (approximately 10 minutes for Bitcoin). Due to this feature, the transaction speed increases. Contrariwise, one of the negative sides of the shorter blocking time is that the struggle of the miners for the next block decision is only intensified. Such competing blocks are also called “blocks without a parent” (these blocks are not included in the main chain of blocks).

Uncles were created so that the miners could receive a well-deserved reward for including blocks without parents in the main chain. Uncles included by the miners in the main chain should be “valid”: they must be child objects in the sixth or earlier generation of the current block. Per exemplum, after the sixth generation, such child objects cannot be included in the main circuit as blocks without a parent: later transactions can negatively affect the operation of the system globally. For uncles, you will receive a reward less than the inclusion of a full block. Nevertheless,

this should not detract from the attempts of the miners to include such blocks without a parent and receive their deserved reward.

Block headers

It has already come up that each block has a header, but we have not really figured out what it is? Block header – is a part of the block that consists of:

parentHash - is the header hash of the parent block (actually, due to it the block gets into the chain of blocks)

ommersHash - hash of the current list of uncles

beneficiary - the account address which receives reward for this block inclusion

stateRoot - the root node hash of the trie state (it was previously mentioned that the trie state is stored in the header, thereby simplifying the state approval process for lightweight clients)

transactionsRoot – the root node hash of the trie containing all the transactions that are listed in this block

receiptsRoot - the root node hash of the trie containing payment information for all transactions listed in this block

logsBloom - Bloom Filter (data structure) consisting of the information contained in the logs

difficulty - complexity level of the current block

number - the number of the current block (the genesis block has a number equal to zero, the block number is incremented by one for each subsequent block)

gasLimit - current gas limit for the current block

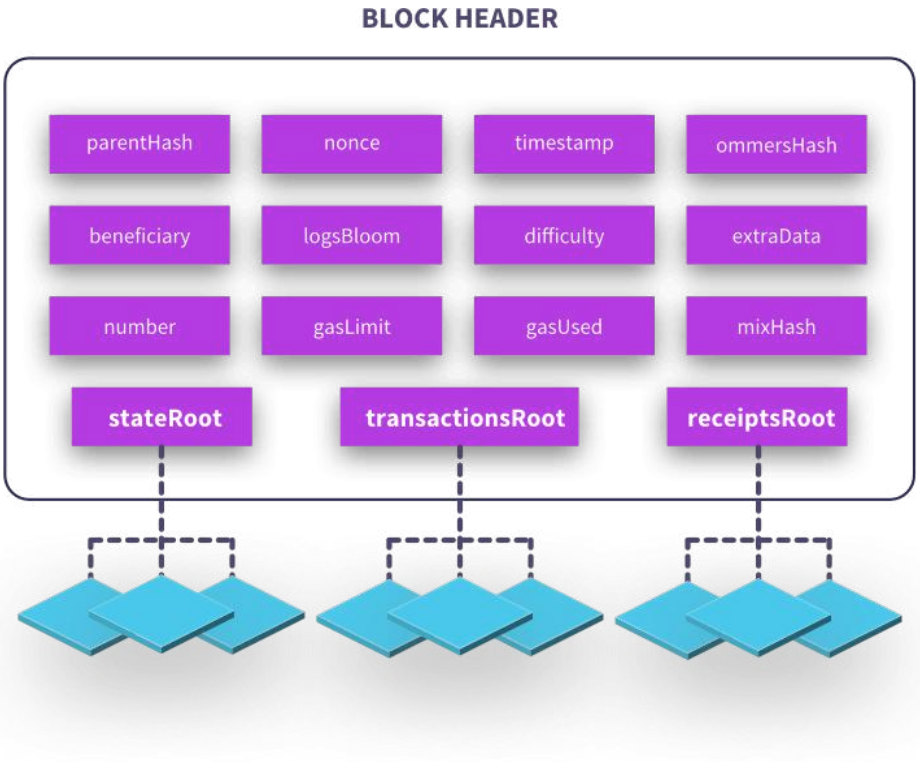
gasUsed - total amount of gas used for transactions in the current block

timestamp – tickmark used to create the current block

extraData - additional data related to the current block

mixHash - hash that, in combination with a nonce element, states that enough computations are performed for the current block

nonce - hash that, in combination with a mixHash, states that enough computations are performed for the current block.



It bears noting, that each block header contains three trie structures for:

- state (stateRoot)
- conducting transactions (transactionsRoot)
- receiving information about payment (receiptsRoot)

These structures of the trie are nothing but the Merkle Patricia trie, which was described above. In addition, there are several terms for this definition, which are intriguing.

Logs

Yocoin platform provides the ability to log, the purpose of which is to record information about various transactions and messages. In addition, it is also possible for a contract to openly create an entry in such a log using the "event" announcement that you want to record.

Log includes:

- the registrar account address
- a series of tasks that display the various events performed for the current transaction
- any data that is relevant to these events

Logs are stored in Bloom Filter by virtue of which it becomes possible to effectively store an infinite amount of data.

Receiving information about payment

The records stored in the header come from the information contained in the log, which refers to the payment transaction data (or check). Just as you receive a check when you purchase items in the store, Yocoin creates a similar check for each transaction. It means that each check contains information about the current transaction. The check includes:

- block number
- block hash
- transaction hash
- total amount of gas used for transaction performing
- total amount of gas used to perform the current transaction, for a specific block
- logs created after transaction performing
- other data

Block complexity

Block complexity is the concept used to ensure consistency of time, which is necessary for the validation of blocks. For the genesis block, the complexity is 131,072 units. To calculate the complexity of any of the blocks, a special formula is used. In case when the validation of one of the blocks occurred more quickly than, for example, the validation of the subsequent one, the protocol used in Yocoin increases the complexity of the last one.

The complexity of the block also affects the nonce - hash, which is required during the block display, while security validation algorithms are used for this purpose.

Nonce, the dependence of one parameter, the block complexity, on the other, is presented in this formula:

$$n \leq \frac{2^{256}}{H_d}$$

where Hd - is block complexity.

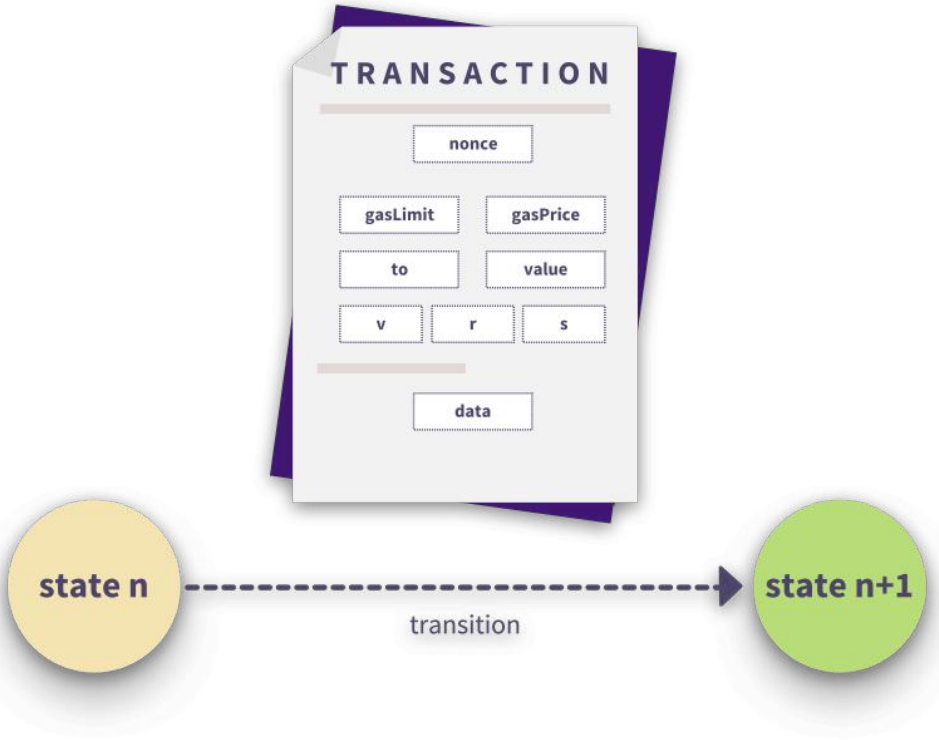
The only way to determine the nonce parameter, which will meet the formula conditions, is to use the sanity check algorithm to find all its possible values.

The expected time for finding all the values that correspond to this condition is the block complexity. Then we can conclude: the greater the block complexity, the more difficult it is to find the nonce parameter, and thus, the more difficult it is to validate the block, resulting in an increase in the time required to validate subsequent blocks. Hence, based on the value obtained during the determination of the block complexity, the protocol assigns how much time is needed to validate the current block.

In case when the time required for the validation of the block is less than expected, then the protocol understates the complexity of the current block. Thus, the time required for the block validation is set automatically to continuously match the current parameters (15 seconds on average).

Conducting transactions

Onward, let's talk about the most difficult part of the protocols used in Yocoin - conducting transactions. Imagine that you have set up a transaction on the Yocoin network. And what do you think will happen with the Yocoin state, when conducting your transaction?



First of all, any transaction must meet certain requirements, so its execution is not canceled, viz:

- transactions must meet the RLP requirements. RLP - Recursive Length Prefix, a data format that is used to encode inclosed binary data arrays. The RLP format is used to organize objects in Yocoin
- valid transaction signature presence
- valid nonce presence. Be reminded, nonce is the number of transactions sent from the current account. In order for this value to be valid, it must match the value of nonce for the sender's account
- transaction gas limit must be equal to or greater than the specified amount of gas. The specified amount of gas includes:

1. a predefined cost equal to 21,000 units of gas required to perform a transaction

2. gas charge used to send transaction data (4 gas units for each data byte or code equal to zero, and 68 for each non-zero data byte or non-zero code)

3. additional 32,000 gas units, if the transaction is related to the contract conclusion

- sender's current account balance must contain enough ETH to cover the "upfront" cost of gas, which the sender undertakes to pay. The upfront cost of gas is calculated as follows: the gas limit is multiplied by gas price for the current transaction, resulting in the max gas cost. Further, to the max cost, the total amount of gas that is forwarded from the sender to the recipient is added.

When you have met all of the above requirements, you proceed to the next step.

First of all, the value of the upfront gas cost is deducted from the sender's account, and the nonce indicator of the sender is increased by 1. After that we can calculate the remaining amount of gas using the following formula: of the total amount of gas needed to conduct a transaction, we deduct a given amount of gas.

During the current transaction, Yocoin traces the "sub-state". The sub-state is necessary for recording the information that was collected during the current transaction. Such information will be required immediately upon completion of the transaction and contains:

Self-destruct set: set of accounts that will be deleted after the transaction is completed

Log series: archived and indexed checkpoints required to run the virtual machine code

Refund balance: the amount that must be returned to the sender when the transaction is completed. It has already been mentioned that the use of the storage provided by Yocoin costs a certain amount of money, and this money is returned to the sender after he ceases to use such a storage. The Yocoin system stores information about the use of the storage and the return of funds for its use to the sender. After that, various necessary computations are performed for transaction conducting.

After all the steps necessary for the transaction conducting have been done (provided that all the above requirements have also been met), the transaction status is completed, and the amount of unused gas that must be returned to the sender is counted.

After the (successful) transaction is handled and the gas is returned to the sender, the following occurs:

- a certain amount of yoc used to purchase gas is sent to the miner
- gas used for the transaction is recorded in the gas count unit (this block is used to store information on the total amount of gas that was used to conduct all transactions in this block, and also this block is used during validation)
- all account information contained in the self-destruct set section is deleted

In the next section, you will learn more about the difference between the transactions associated with creating contracts and sending messages.

Creating contracts

As mentioned above in Yocoin there are only two types of accounts: contract accounts and externally owned accounts. When you meet the term “transactions associated with the creation of a contract”, you should know that the purpose of such a transaction is a new contract account creation. To create a new contract account, first of all we have to declare the address of the created account using the special formula. After that, a new account is created. To perform this operation, you must perform a number of actions:

- put zero in nonce
- adjust the balance of your account equal to the payment for the transaction (when the sender is ready to send a certain amount of yoc as payment for the transaction)
- count the amount of payment that is transferred to the account balance from the sender account
- specify that the storage is no longer used by you
- customize the hash code of the contract as the hash of an empty line

At the moment, when we started to create a new account, we, in fact, already created it with init-code, which is automatically sent at the beginning of the transaction (see section “Transactions and messages”). There are few scenarios during the init-code running (for example, updating the storage for an account, creating another account for the current contract, sending a message, etc.).

After the system performs the code designed to create a new contract, gas enters the game. You will not be able to conduct a transaction if it requires more gas than is stored on your balance. If, despite a restriction, you attempt to conduct a transaction, you will receive a message about the lack of gas, the system will automatically close straight after. At the same time, if the transaction was terminated due to a lack of gas, then you will be transferred to the stage preceding the transaction. And what's the main: the recipient will not be refunded the amount of gas that was spent before the leakage of its shortage.

Withal, if the sender has placed a certain amount of yoc for the current transaction, this amount will be returned even if the contract creation fails.

If the initialization code was performed successfully, then the funds required to create the contract must be tender by the creator. This amount also includes the cost of using the storage, which increases proportionally with the increase in the size of the code created for the contract. But when that the creator does not have enough funds to carry out this operation, the transaction ceases due to lack of gas and the consequences will be the same as those given above.

If everything went smoothly and we did not receive a report of gas shortage, then all the gas that was not used for this transaction is returned to the sender.

Messages

Broadly speaking, the operation of sending a message is quite similar to the creation of a contract, not taking into account some minor differences.

To perform this operation, the use of the init-code is completely unnecessary, because as a result of its execution no new account is created. However, such an operation may require input data, but only if data was transferred by the recipient as a result of the transaction. After message sending, a new block that contains the output information is available, the use of which occurs when the operation is repeated.

Just as in the case of the contract creation, if the sending operation execution was interrupted due to lack of gas or an invalid transaction (f.ex., due to a stack overflow error, incorrect jump address, incorrect command), the amount of gas used for this operation is not returned to the call initiator. Conversely, all unused gas is also written off from its balance, and the state of the system returns to the point preceding the balance transfer operation.

Until quite recently in Yocoin it was impossible to interrupt or suspend the execution of transactions without loss of gas provided for such purpose. For example, imagine the situation when you are the initiator of contract creation and the error occurred because the initiator of the call did not have the right to perform any of the transactions. So, in Yocoin previous version, before updating the platform, in this situation, all the gas remaining in your account would be removed, while the sender would also not get his gas back. But for today - you have the opportunity to suspend the execution of contract creating operation and return the system to its genesis state without losing the remaining gas on your account. Thus, if the exit from the transaction occurred as a result of the suspension of its execution, then the unused gas is returned back to the sender.

Performance model

In previous sections was told how the transactions are conducting. Now let's examine what happens in the VM (Virtual Machine) at the moment of transaction conduction.

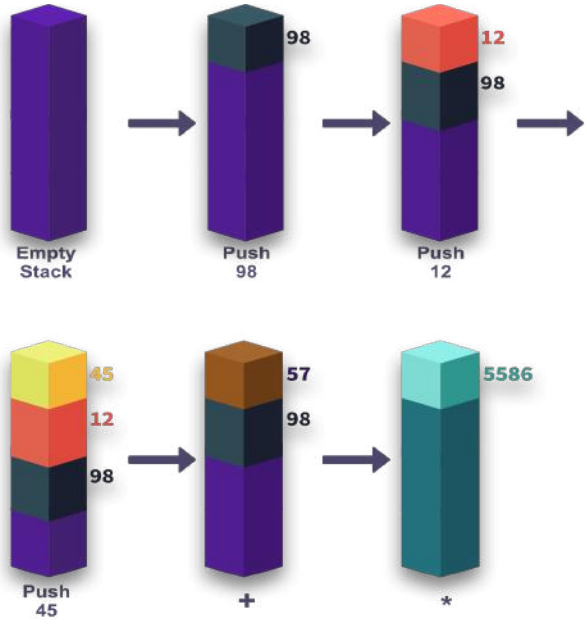
The part of the protocol that performs the transaction processing in the Yocoin operating system is called the Yocoin virtual machine (VMY). As mentioned above VMY is a Turing machine. The one and only difference between VMY and a typical Turing machine is that the first one requires virtual “gas”. Thus, all the computations that can be performed in VMY are somehow limited by the amount of “gas” circulating in it.

In addition, VMY has all the features of the stack architecture. Stack machine – a computer which deploys LIFO algorithm.

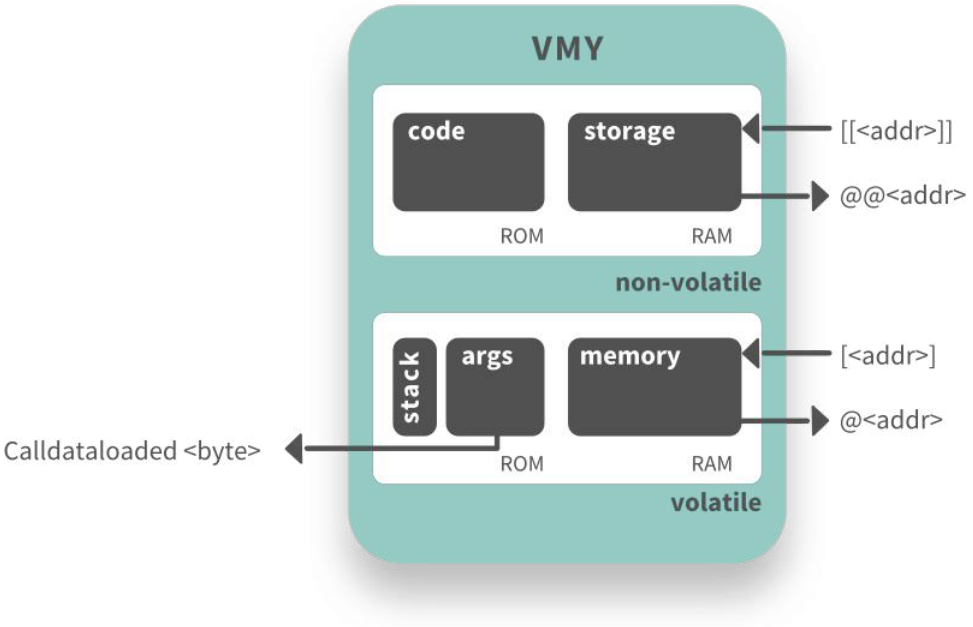
The size of any stack element in the VMY is 256 bits, and the max stack size is 1024 bits.

The certain amount of inconstant memory is foreseen for VMY. The elements are stored as arrays of bytes with reference to words there.

VMY also has a specific storage area. Unlike the memory space, such storage (or storage area) does not change and is a part of the state system. In VMY, the program code is stored in a separate virtual ROM, access to which can only be obtained through specific instructions. From this point of view, VMY differs from the typical von Neumann architecture



where the program code is stored in the computer's memory.



VMY also provides its own special language - byte-code VMY. When a programmer creates a smart contract that will run on the Yocoin system, this usually allowable with a high-level language, Solidity. After writing this code, we compile it into the VMY bytecode so that VMY can understand the command we wrote.

Let's move to directly to completing activity.

Before performing a specific computation, the processor must ensure that the following information is valid and available:

- system state
- information on the gas amount sufficient to perform the required operation
- address of the account to which the executable code belongs
- address of the transaction sender - initiator of the current transaction
- operation input data
- amount of Wei that must be sent to the account as a result of current operation
- information about the machine code being executed
- block header information for the current block
- depth of the current message or the contract creation

Immediately before the execution of the program, the system memory is completely empty, and the instruction counter is zero.

PC: 0 STACK: [] MEM: [], STORAGE: {}

Then the recursive execution of the transaction begins in VMY: the computation of the system state and the machine state for each cycle. System state is Yocoin global state. Machine state includes:

- available amount of gas
- P-counter
- memory content
- active number of words in memory
- stack content

Stack elements are added or removed from the left edge of the code snippet. For each cycle, a certain part of the remaining gas amount is taken away, while the command counter is increased.

There are three possible versions of the cycle termination:

1. The operations performed by the machine reach an exceptional state (f. ex., due to lack of virtual gas, incorrect instructions, insufficient number of stack elements, exceeding the size of 1024 bits, incorrect assignment of JUMP/ JUMPI) and thus the process of the operation is suspended.
2. The sequence of actions proceeds to the next cycle.
3. The operations performed by the machine reach a logical conclusion (process completion).

When the computations performed by the machine reach a logical conclusion, rather than an exceptional state, the result is that the machine produces the resulting state, as well as information about the remaining gas and the resulting output data.

We just have had an idea of the most complicated and intricate part of Yocoin. Do not worry if you do not understand something completely: there is no need to delve into every little thing and understand all the processes taking place in this system, well, unless you are going to study it and work at a sufficiently deep level.

Final block designation

Let's finally figure out what happens to blocks transaction during their final designation.

“Final designation” can occur in two ways, depending on whether we create the block or it is already been created. In the event that we only create a block, the final designation means the process of mining the current block. On the other hand, when block is already been created, such a definition means the process of validating the current block. In both of the cases above, four conditions must be met for the final designation of the block:

- Validation of uncles: each block of uncles that is in the block header must have a valid block header and be the sixth child object of the current block.
- Transaction validation: the gasUsed value for the current block should be equal to the value of the total amount of gas used to conduct all transactions listed in this block.
- Purpose of payment (only in case of mining): beneficiary is assigned 10 yocs for the mining of each block. Moreover, for each uncle, the current block is assigned a payment of additional 1/32 of the total payment for the current block. And finally, the beneficiary of the block of uncles is also assigned a payment in the form of a certain amount, for the determination of which there is a special figure.
- State verification and nonce values: to perform this procedure, you need to ensure that all transactions are performed, as well as changing the resulting states. After that you will also need to specify a new block after the payment for this block has been sent. The verification process occurs by comparing the final state with the state of the trie stored in the header.

Mining aimed at proving work

In section “Blocks” the block complexity concept was described. The algorithm, thanks to which the notion of the block complexity arose, is the Proof of Work (POW).

POW algorithm used in the Yocoin system is called Ethash. This algorithm has the following formula:

$$(m, n) = PoW(H_{n'}, H_n, d)$$

where m is mixHash, Hn is the new block header (nonce and mixHash are not included here because these values should be calculated), Hn- block header nonce, d- DAG data set.

In section “Blocks” were also described multiple definitions, provided for the block header. These include values such as mixHash and nonce.

Reminding once again:

mixHash - hash that together with the value of nonce confirms that a sufficient number of computations have been performed for the current block.

nonce - also represents a hash that, together with the mixHash, confirms that a sufficient number of computations have been performed for the current block.

Consequently, PoW algorithm is needed to calculate the above values.

It is a rather complex task to explain exactly how mixHash and nonce are calculated using the PoW function. But in short, the following happens:

The value of “seed” is calculated for each of the blocks. To count each of the seeds, there is its own “interval”, equal to 30,000 blocks. For each interval, the seed is a hash equal to a series of 32-byte zeros. For each subsequent interval, a specific hash is provided. Using this seed, the node finds the value of a pseudorandom “hash”.

Such a hash plays a significant role, because with its help we can better understand what the “lightweight nodes” are, discussed in the previous sections.

The goal of lightweight nodes is to enable some of the nodes to efficiently check certain transactions without having to store the entire Blockchain dataset. Lightweight node can perform the transaction validation, using only this hash. This happens due to the fact that the hash can re-create the block it needs for verification.

Using this hash, the node can create a DAG dataset, where each element depends on a small number of randomized pseudohash elements.

Every entry-level miner must create his complete dataset for a start. In system, a separate dataset is stored for each miner and the volume of data continuously increasing.

As an example, miner can take any random parts from the dataset and use them in a mathematical function in order to hash such parts for the mixHash. Such a miner can always set the value for mixHash until the source data is received as nonce value. When this condition is met, the nonce value will be considered valid, and the block can be added to the chain.

Mining as protection mechanism

Globally, the purpose of PoW is to cryptographically prove that certain computations were aimed at obtaining a certain result (nonce values). It happened that there is no other way of finding a nonce, whose value does not exceed a certain limit, except with the help of enumerating all possible options up to finding the required one. The output data distribution for the permanently

used hash of functions occurs evenly. Thus, we know for sure that the time necessary to find the nonce value clearly depends on the complexity threshold: the higher the complexity threshold, the longer the search for the necessary nonce value will take place. The PoW algorithm represents the concept of complexity used in this Blockchain.

So, what does it mean “secure Blockchain”? The answer is pretty simple: a secure Blockchain is a Blockchain that will be trusted with absolutely ALL USERS. As it was mentioned above, if there are more than two chains in Blockchain then it is quite logical to assume that users will not feel confident while working with Blockchain, since no one can tell which of the presented chains is valid.

Exactly for this purpose the PoW algorithm is applied: ensures the unity of the Blockchain, preventing the creation of other chain of blocks that can affect the transaction history (f.ex., creating unreal transactions or deleting, or modifying existing ones). Thus, for an attacker to be able to validate his blocks first, he will have to constantly determine the value of nonce, and do it faster than all other network users (see section GHOST protocol). Of course, for the attacker this method would not be feasible, unless the majority of miner’s network sources are at his disposal - such a scenario is known as an attack of 51%.

Mining for finance distribution

Apart from the fact that the PoW algorithm ensures the Blockchain safe performance, it also distributes the reward to those users whose computations have been used to ensure security. As mentioned above, miners receive a reward for block mining, as well as:

- reward of 10 yocs for the “winning” block
- consumed gas cost as a result of the transaction in the block
- additional reward for including uncles in block

To ensure the consistency of the PoW method, necessary to guarantee security and reward distribution, Yocoin constantly adheres to the two principles presented below:

- First of all, to attract as many users as possible to the platform. In other words, the use of this platform should not cause the user any difficulties: it should not use any supercomplex algorithms or unknown hardware. In addition, the reward distribution process should also be clear and simple for anyone who is willing to spend some energy used by his computer to get a few cherished yoc units.
- Second of all, do not allow disproportionate distribution of rewards and other resources for any particular node: any node for which a disproportionate allocation of resources occurs will have a huge impact on the definition of canonical Blockchain, which negatively affects the security of the system as a whole.

For example, in the Bitcoin system, there is a problem with the implementation of the above two principles: its PoW algorithm uses the SHA256 hash function. The problem is that the solution can be much easier in case of using special hardware - ASICs.

In order to prevent such punctures in Yocoin, a special PoW algorithm with sequential memory (Ethash) is used. The structure of the algorithm is constructed in such a way that the computation of the nonce value requires the use of a large amount of memory and high connection throughput. The requirements for having a large amount of memory means that for a computer with the usual amount of memory it will be very difficult to parallely compute several nonce values. As for the requirements for high bandwidth, even for an ultra-fast computer, detecting several nonce values at the same time will become a challenge. Thus, due to such features of this system, the risk of risks centralization is reduced and, in addition, more even conditions are created for the operation of various nodes that perform verification.

References

1. Ethereum. Ethereum. https://ethereum.org
2. Gavin Wood. ETHEREUM: A SECURE DECENTRALISED GENERALISED TRANSACTION LEDGER. http://gavwood.com/paper.pdf, Feb 2015.
3. Satoshi Nakamoto. Bitcoin: A Peer-to-peer Electronic Cash System. https://bitcoin.org/bitcoin.pdf, Oct 2008.
4. Nick Szabo. Formalizing and Securing Relationships on Public Networks. http://szabo.best.vwh.net/formalize.html, Sep 1997.
5. Naval Ravikant. The Bitcoin Model for Crowdfunding. https://startupboy.com/2014/03/09/the-bitcoin-model-for-crowdfunding/.
6. C. Detrio, “Smart markets for smart contracts,” 2015. [Online]. Available: http://cdetr.io/smartmarkets/.
7. P. Snow, B. Deery, J. Lu, et al., “Factom: Business processes secured by immutable audit trails on the blockchain,” 2014. [Online]. Available: http : / / bravenewcoin.com/assets/Whitepapers/ Factom-Whitepaper.pdf
8. A. Swartz, “Squaring the triangle: Secure, decentralized, human-readable names,” 2011. [Online]. Available: http : / / www . aaronsw . com / weblog / squarezooko.
9. R. C. Merkle, “Protocols for public key cryptosystems,” in IEEE Symposium on Security and Privacy, 1980
10. “Schema.org schemas,” 2016. [Online]. Available: http://schema.org/docs/schemas.html.
11. Smart property: https://en.bitcoin.it/wiki/Smart_Property
12. GHOST: http://www.cs.huji.ac.il/~avivz/pubs/13/btc_scalability_full.pdf
13. Simplified payment verification: Simplified payment verification
14. Colored coins whitepaper: https://tinyurl.com/coloredcoin-whitepaper