**Yuamble Yellowpaper**

Ivanov Alexandr (@iaa2005)

yuamble.org

**Abstract.** This document is devoted to the description of a new type of blockchain — Yuamble. Blockchain is a distributed registry that stores transactions, data on smart contracts (to be disclosed in detail later), as well as other data of users and participants of the blockchain network. This document will talk about algorithms that will reduce the size of fees on the network, as well as make the blockchain post-quantum.

**Introduction.** The Yuamble blockchain is very similar to the Bitcoin and Ethereum blockchain. It performs the same main functions, such as storing data about users and their transactions. But Ethereum has expanded the possibilities of using blockchain – this is the use of smart contracts.

Smart contracts are a computer program that monitors and ensures the fulfillment of obligations. The parties prescribe in it the terms of the transaction and sanctions for their non-fulfillment, put digital signatures. A smart contract independently determines whether everything has been executed and makes a decision: to complete the transaction and issue the required (money, shares, real estate), impose a fine or penalty on the participants, close access to assets.

But often the blockchain network consisting of these smart contracts is very heavily loaded. Network participants who send signed transactions to the pool must pay huge fees to reduce the processing time of this transaction, and so that the transaction gets into the block — the main part of the distributed registry as soon as possible. To solve this problem, we are making some changes to the protocol — the blockchain algorithm.

1. ***Using DAG technology.*** A DAG is a directed acyclic graph. Instead of Blockchain technology, this technology will be used. It will allow the registry to process transactions quickly and simultaneously add several blocks from transactions to the chain. [See the file](https://files.ifi.uzh.ch/CSG/staff/Rafati/Simon_MBM_IOTA.pdf).
2. ***Using the digital signature of Winternitz.*** The Winternitz signature (W-OTS, Winternitz One Time Signature) is an improved algorithm for the digital signature of a Lamb port. This algorithm signs transactions quickly. It is post-quantum, i.e., quantum computers and their algorithms will not be able to find a private key to a public one, forge a transaction signature and pass it off as the real account owner. [See the file](https://eprint.iacr.org/2011/191.pdf).
3. ***Make processing of smart contracts without fees.*** Most blockchains process transactions only with the payment of a commission. Due to the payment of the fee, the number of participants in the network who sign transactions and send them to the pool decreases over time, because the size of the fee increases with the demand for transaction processing. We made the main decision — to remove fees on transactions and calls of smart contracts under one condition: if the coefficient ***k*** is the probability coefficient of looping, it will be approximately zero.

This coefficient shows how secure the code written by the creator of the smart contract is for nodes participating in the Yuamble network. Since in the presence of malicious code, Yuamble virtual machines (YVM, an analogue of the [Ethereum Virtual Machine EVM](https://ethereum.org/en/developers/docs/evm/)) can go into a resource-draining endless cycle. There are several ways to provide a way to terminate the contract from the outside and avoid entering into a resource-draining infinite loop:

1. *Turing incompleteness:* Limited functionality will not allow jumping and/or loops. Therefore, the smart contract will not be able to enter an infinite loop.
2. *Step and Cost Meter:* The program can simply track the number of commands executed, and then shut down after completing a certain count of steps. Another method is a counter. Here contracts are executed with prepayment. A certain amount is required to complete each instruction. If the fee paid exceeds the prepaid fee, the contract is terminated.
3. *Timer:* if the execution of the contract does not meet a certain deadline, then it is terminated forcibly.

We will use *Turing Incompleteness* and a *Timer*, because the *Step and Cost Meter*, as in Ethereum, will not work if we have transaction processing without [fees and gas](https://ethereum.org/en/developers/docs/gas/).

The entire protocol will be written in Rust, C++. Rust provides great advantages. It is convenient to write code on it and connect important libraries and modules for working with the network, digital signatures and smart contract processing.

*We believe that this project will provide huge opportunities in the field of decentralized finance, banking, programming and cryptography. This project will overestimate the possibilities of the blockchain and make the world of censorship stable and private for the participants of the Yuamble network.*