Bruno Coin

A Smiling Cryptocurrency

Whitepaper V1.0

Introduction

There are many realizations of p2p electronic currencies - Bitcoin, Etherium and others. All of them were well received by the general public and professionals, thanks to a multitude of advantages - decentralization, the absence of a single emission center, the publicity of transactions and so on. However, the publicity of transactions was both a plus and a minus - not always the payer wants to publish the fact of payment, and the payee to advertise its receipt. The anonymity of payments, which had to be used for gold transactions from time immemorial, and due to paper cash, allowed to accumulate capital without undue attention from criminal and / or unscrupulous governments. However, already at the time of occurrence of paper cash, there was a problem of issue - unscrupulous rulers were all tempted by the opportunity to release a lot of unsecured notes to the market, thus robbing their own people for personal purposes by inflation. The transfer of paper cash to the electronic form of bank deposits ended as well and with anonymity. And due to poorly solved security issues, information about the contributions of citizens is practically not protected. And the volume of the emitted currency is known in fact only to a narrow group of persons, they are also beneficiaries of this issue. Managing the emission of the currency, these narrow groups of people manage the distribution of capital on a national scale, and in the case of world reserve currencies - across the whole planet, if desired, redistributing liquidity into their own pockets through artificially induced crises and inflation. In parallel, through pocket governments, regulating the rules of the game on the market in such a way that a simple person or even a capital owner is simply obliged to invest his capital in the markets under other people's rules, giving up to half or more profits to those who stand at the top of this pyramid.

All this injustice, as well as a deficit of investment funds caused by the 2008 cryses, created the appeal of the Crypto-currency as a means of settlement and investment. Bitcoin was a pioneer and many shortcomings were discovered quite quickly in it. Some of them have been fixed with various forks, like a hash vulnerable to collisions, non-anonymity of transactions (and their slow speed), block size limits and so on. Some of the shortcomings have already been fixed in different crypto-currencies, some are still waiting for their hour.

But some serious shortcomings are still hidden from the general public and they still have to be resolved (although particular solutions already exist). It's about the so-called "quantum apocalypse". It is on his decision (although not only him) we suggest to sharpen your and your attention

1.2. Quantum Apocalypse

The appearance of quantum computers capable of solving a number of problems at a fundamentally higher level of efficiency, in comparison with classical computers, was predicted as early as the 1980s and 1990s. But only at the end of the 00's there were real prototypes. And in the middle of the 10th there appeared practically ready-made variants of quantum computers suitable for solving practical problems. One such task is to search for a private key based on a public key - a task on the complexity of which all modern cryptography is based. Not for all versions of cryptography, quantum computers give an advantage, but unfortunately for the most common version of RSA there is already a so-called "Shore algorithm", which allows to solve the problem of finding a private key public in a relatively short time. For the second variant of signatures, ECDSA and others similar to it, there is also the possibility of hacking, although in practice there is still no algorithm for it. Vulnerability to quantum computers is recognized by the Bitcoin authors themselves: https://bitcoin.org/en/faq#is-bitcoin-vulnerable-to-quantum-computing
"Yes, most systems are relying on cryptography in general, including traditional banking systems. However, quantum computers do not exist. In the event that quantum computing could be an imminent threat to Bitcoin, the protocol could be upgraded to use post-quantum algorithms. Given the importance that this update would have, it can be safely assumed that it will be highly reviewed by the developers and adopted by all Bitcoin users. "

Apparently, the problem is serious enough and is recognized by all present. With one exception - the time when quantum computers appeared on the market.

https://motherboard.vice.com/en_us/article/7xwz7b/53-qubit-quantum-simulator-computer

Two teams of researchers have published papers in the nature of how they were able to create unprecedented quantum simulators. This means that these experimental quantum computers are on the cusp of being able to model physics that is too complex even for the most powerful conventional supercomputers.

This is already enough to crack most of the implementations of RSA. Thus, even everyday communications are not in danger, the updates to the protocols are likely to come out in the next year or two. The greatest danger is hung over the crypto-currencies and, above all, Bitcoin. After all, the only thing that confirms your fact of possession of funds in the detachment is a private key.

Of course, they know about this problem. But just as everyone knows how difficult it is to make changes to the block system, in which backward compatibility is lost, the so-called hardcore. In view of the fact that the capitalization of the largest crypto currency is already estimated in hundreds of billions of dollars, the temptation to create its own fork, its own version of the crypto currency at the time of its hardcore, is very high. If successful, you can earn tens and hundreds of billions of dollars, with little or no effort. This is exactly what happened in 2015 c Etherium, and in 2017 with Bitcoin. Both these events severely undermined the capitalization of these currencies, causing an outflow of funds to forks and undermining investor confidence.

It is obvious that when the crypto currency transition begins to encryption algorithms that are stable to quantum computing, the same will happen, but on a larger scale. Quantum apocalypse will cause a crisis crisis, during which the advantages will be those crypto-currencies that will withstand the quantum apocalypse. This is the currency we are announcing.

1.3 Post-Quantum Encryption and Digital Signature Algorithms

Since the late 1980s, a number of so-called post-quantum algorithms that are stable to quantum computing or at least difficult to crack and quantum computers have been proposed. In particular, hash trees, on which the integrity of the blockade is built. In this regard, the blockade is invulnerable, - a quantum computer will enable you to withdraw money from someone else's wallet, but will not allow you to change the existing transaction history. Unfortunately, this algorithm is bad enough for

forming a digital signature due to its unwieldiness and poor scaling (although in theory one can also use it).
In the last few years, a number of other, more convenient algorithms have appeared. Most of them are assembled together by the Open Quantum Safe organization (https://github.com/demlabsinc/libdap/issues), some are already implemented in Google Chrome, in some Microsoft products, something is recommended by the IEEE as a new standard. Also, there is already a post-quantum algorithm for the formation of Picnic ring digital signature, when using which it is impossible to determine which of the network participants signed this transaction.
It is on the combination of the algorithms OQS and Picnic that we are going to base the protection of funds in our new crypto currency.

1.4. Transactions, balance between the means of payment and the means of accumulation, multiple blocking

There is one more aspect of crypto currency, which is problematic - transactions, their speed and cost. The cost of transactions grows with the complexity of the network, which begins to "inflate" the active mining of tokens. However, without active mining there will be no transactions, and the attractiveness of tokens will decrease.
Part of the problem is solved by removing the limit on the block size, but only partially. The cost of the token itself is also growing and although it is only a small part of the cost of transactions - it is its limit. With the growth of the economy on the one hand, computing power with another and capitalization, third-party crypto currency will sooner or later come a situation where the cost of one token is equal to the market value of transactions that on average come to this very token.
On the one hand, this will make transactions sufficiently protected because of their high cost. Perhaps in the future, Bitcoin's transaction itself will be a beautiful application to a large amount of investment and serve as a certain quality guarantee.
On the other hand, it is the problem of transactions that has generated and continues to generate more and more new projects. That however also creates some inconvenience, it is necessary to keep various cryptotices for daily payments and personal savings. And since usually a good investment vehicle due to the rise in

price of tokens in the future can not have cheap transactions, there is still no single solution in this area.

The DapCash project offers its solution to this problem, which, like the previous solutions, is supposed to be flexible and variable - multiple blocking, in which blocks are combined into different chains with different types of tokens, but which can contain each other's transactions. Calculation of the balance of the wallet will be made immediately for all the detainees. Different blockhouses will have different algorithms for forming a block, different block generation times, different complexity. In the future, probably the emergence of new types of tokens.

2. Features of the implementation of DapCash

The main ideology of implementation is variability and extensibility. We must be prepared for a situation that in 10 years will find vulnerability in the algorithms, in the structure of the blockade, they will flood the network with a lot of transactions, try to attack 51%, and so on.

2.1. The variational post-quantum signature

Any digital signature must be variable - that is, it can be implemented in the form of different algorithms. It can be ring, allowing to make anonymous transactions, it can be multiple - signed by several participants at once. It can be of different types. If tomorrow you find a vulnerability in the security protocol, the next day the update of the software should add a new id for the new secure protocol.

2.2. Proof-of-Work function

For different blockages in the DapCash can also be different algorithms Proof-of-Work. For the "golden" block, at number 0, Proof-of-Work is supposed to be similar to what exists in Monero - calculation of the hash tree, where the hash function is derived from AES, like SALSA. In this case, SSE2 / SSE3 registers in the processors will give a very big gain in performance. Plus, the size of the tree should be about 3-4MB, so that the cache of top processors crawls into L3.

The positive result of mining, like everyone else, is a hash, in which the last N digits are "special."
"Feature" for different detachments can be different. The "gold" is a classic zeros. The "silver" is any other figure. Thus, the "gold" to "silver" will be fixed is tied approximately 1 to 9 and it is in this ratio that tokens will be taken into account. Subsequent tokens, "copper", "iron" or even "wooden", if there is a need for their release, may already have a more flexible system of relationships and it will be determined already by the network needs - the average number of pending transactions per block

2.3. Transactions

The peculiarity of multiple blocking is that transactions must be considered immediately in all. Transaction of the "golden" token can be described both in the "gold" block and in the "silver" or in any other.
Due to the variability of blockers, types of records and addresses, the size of the transaction can also vary greatly back and forth, especially in the case of anonymous transactions. Since the size of the transaction will be taken into account in the succession formula for placing transactions in the block, larger ones will imply a larger commission for the transfer. However, in view of the multiple blockage and the possibility to increase the total number of units that go online, this all has very little to affect the speed and cost of transactions - they will necessarily remain low.

2.3.1. Anonymous transactions, the CryptoNote algorithm

The concept of anonymous transactions was borrowed from the CryptoNote community: https://cryptonote.org/ but we did not take the proposed implementation. At least in view of the fact that the ring signature proposed by them is based on classical cryptography. We have already integrated the implementation of post-quantum annotation Picnic - an algorithm, as well as based on a zero-knowledge confirmation. as described in the following article Melissa Chase and David Derler and Steven Goldfeder and Claudio Orlandi and Sebastian Ramacher and Christian Rechberger and Daniel Slamanig and Greg Zaverucha,
https://eprint.iacr.org/2017/279.pdf

The rest of the elements of our implementation of the CryptoNote concept are not fundamentally different, except for any restrictions on the number of one-time signatures and their respective chains. For the transaction, just one additional data block is added to the signature block format transaction, between the Input and Output elements of the transactional section of the block. Thus, it acts as an Extra unit in CryptoNote and at the same time is a more universal and transparent unit, allowing over time to create various hybrid schemes for generating digital signatures and transaction checks.

3. Technical details of the implementation

The basis for the technical implementation of the block is the DAP framework, specifically the following libraries

- https://github.com/demlabsinc/libdap
- https://github.com/demlabsinc/libdap-stream
- https://github.com/demlabsinc/libdap-chain
- https://github.com/demlabsinc/libdap-server

3.1. Blockchain format

Block is the set of blocks. Nothing else but the information inside the blocks themselves characterizes the block system - all headers, caches, distributed key-value data are only secondary data. Primary are only those that are inside the block.

3.1.1 Block Format

The block consists of a header, which includes general information and a varied set of sections. The maximum block size can be limited. In the case of the "golden" block this limit is 128 kilobytes, the "silver" block will be limited to 10 MB of size, the size of the copper block of limits will not be available and will be limited only by the network bandwidth and transaction costs.
The header of the code consists consistently of the following data (all little endian):

Block header

4 bytes	**:**block signature, for gold and silver blocks 0xDA05BF8E
4 bytes	**:**block format version, signed integer, currently 1
4 bytes	**:**unsigned size of the entire block, including signature, header and sections
32 bytes	**:**hash of the previous block
8 bytes	**:**block creation time
8 bytes	**:**network complexity at the time of generation
8 bytes	**:**nonce - the number that is selected during the mining
32 bytes	**:**HashFusion hash tree root

Then follow the sections of the block, one after another. In sections, there can be transaction details, such as transaction input, output, signatures and intermediate one-time keys.
Each section starts with two bytes that define its type - it can be a transaction, a smart contract code, a set of keys, a message, and so on.

3.2. Authentication Algorithms

Then follow the algorithms of digital signatures, which will be provided by the system at its launch. The maximum number of algorithms is 65535.

kex_rlwe_bcns15:key exchange from the ring learning with errors problem (Bos, Costello, Naehrig, Stebila, IEEE Symposium on Security & Privacy 2015, https://eprint.iacr.org/2014/599)

kex_rlwe_newhope:"NewHope": key exchange from the ring learning with errors problem (Alkim, Ducas, Pöppelmann, Schwabe, USENIX Security 2016, https://eprint.iacr.org/2015/1092) (using the reference C implementation of NewHope from https://github.com/tpoeppelmann/newhope)

kex_rlwe_msrln16: Microsoft Research implementation of Peikert's ring-LWE key exchange (Longa, Naehrig, CANS 2016, https://eprint.iacr.org/2016/504) (based on the implementation of Alkim, Ducas, Pöppelmann, and Schwabe, with improvements

from Longa and Naehrig, see
https://www.microsoft.com/en-us/research/project/lattice-cryptography-library/)

kex_lwe_frodo:"Frodo": Key exchange from the learning with errors problem (Bos, Costello, Ducas, Mironov, Naehrig, Nikolaenko, Raghunathan, Stebila, ACM Conference on Computer and Communications Security 2016, https://eprint.iacr.org/2016/659)
kex_sidh_cln16: key exchange from the supersingular isogeny The Diffie-Hellman problem (Costello, Naehrig, Longa, CRYPTO 2016, https://eprint.iacr.org/2016/413), using the implementation of Microsoft Research https://www.microsoft.com/en-us/research/project/sidh-library/

kex_sidh_iqc_ref:key exchange from the supersingular isogeny The Diffie-Hellman problem (De Feo, Jao, Plot, J. Math. Cryptol. 8 (3): 209, 2014, https://eprint.iacr.org/2011/506), using a reference implementation by Javad Doliskani

kex_code_mcbits:"McBits": Bernard, Chou, Schwabe, CHES 2013, https://eprint.iacr.org/2015/610), key exchange from the error correcting codes, specifically Niederreiter's form of McEliece , using the implementation of McBits from https://www.win.tue.nl/~tchou/mcbits/)

kex_ntru:NTRU: key transport using NTRU public key encryption (Hoffstein, Pipher, Silverman, ANTS 1998) with the EES743EP1 parameter set, wrapper around the implementation from the NTRU Open Source project https://github.com/NTRUOpenSourceProject/NTRUEncrypt)

kex_mlwe_kyber:A CCA-secure module-lattice-based key exchange mechanism (Bos, Ducas, Kiltz, Lepoint, Lyubashevsky, Schwabe, Shanck, Stehlé, Real World Crypto 2017, https://eprint.iacr.org/2017/634)

sig_picnic:based on zero-knowledge proof as specified in the Post-Quantum Zero-Knowledge and Signatures from Symmetric-Key Primitives (Melissa Chase and

David Derler and Steven Goldfeder and Claudio Orlandi and Sebastian Ramacher and Christian Rechberger and Daniel Slamanig and Greg Zaverucha, https://eprint.iacr.org/2017/279.pdf), using the optimized implemenation from https://github.com/IAIK/Picnic

3.3. Formation of a new block

Just like in all other PoW networks, the complexity of forming each new block is adjusted on the basis of the total hash of capacities so that the generation of each new block takes place at the same time. For gold blocks this time is extremely high, ten thousand seconds (10,000). The golden block is confirmed by 10 inclusions in the silver block plus three inclusions in the gold.
Mining of both blocks goes simultaneously, with calculation of probability of that for each one thousand silver blocks there will be one golden

3.4. Transactions

Transaction is one of the types of sections in a block. Its title consists of only two parts:

Transaction Header

8 bytes	**:**locktime (zero) if not used
4 bytes	**:**The size of the subsequent block of transaction elements

Further after the header follows a block of elements, such as input (In), output (Out), public key (PKEY), signature (SIG).
Each of the elements also has its own header and the body following it. Each header necessarily begins with a byte that specifies the type:

Login header (in):

4 bytes	**:**type (0x00)
32 bytes	**:**hash of the previous transaction (0 for the base transaction)
2 bytes	**:**type of signature
4 bytes	**:**signature size
4 bytes	**:**transaction number in the chain

The body of the entrance (in)

The signature body encoded in base58

Out header:

4 bytes:type (0x10)

8 bytes: Number of sent Datoshi (1/10 ^ 9 from the token dongle)

2 bytes:type of signature

42 bytes: destination address

Out body

Not available

Public key header (pkey):

4 bytes:type (0x20)

2 bytes:key type

4 bytes:size of the key

4 bytes: transaction number in the chain to which the key belongs

Public key body (pkey)

The public key body encoded in base58

Signature header (sig):

4 bytes: type (0x30)

2 bytes:type of signature

4 bytes: signature size

Signature body (sig)

The signature body encoded in base58

3.5. Network protocols

The protocol for data exchange is two-stage, so-called session and streaming. The basis of its implementation is the DAP () framework, specially designed for fast and secure connections. The main protocol is assumed to be streamed, with independent transmission units. Due to the peculiarities of the block, the main transport of the streaming protocol proposes to use UDP, while maintaining the possibility of interaction via TCP and even HTTP protocol. The exchange of nodes between nodes by default will be done through UDP, unless of course local network restrictions will allow this. If not, then HTTP or any other available transport may be involved.

The very fact of the operation of a node, especially a large node around a purse with large funds, may be undesirable in terms of publicizing this fact to the state or other structures. Because of this, all communications with the nodes will be encrypted to also complicate the tracking of anonymous transactions by intruders.

3.5.1. Session protocol

The session protocol is implemented in the libdap library (https://github.com/demlabsinc/libdap) and is a classic request-response protocol over HTTP.

The session DAP protocol begins with the initialization of the session key. This is similar to SSL handshake way - the client generates a couple of post-quantum keys, public and private, signs them with their own personal key (advertised once and publicly when registering on the network) and passes it to the server. The server checks the signature of the public key, checks the validity of the data decryption, generates another key pair from its side, assigns it a local identifier and passes it to the client along with its public session key. This step is called ENC_INIT.

Next, with each session request, the HTTP headers are added with the Cookie header, which specifies the key's idem, and the body itself is sent to them encrypted.

3.5.2. Streaming DAP Protocol

The streaming protocol is implemented using the libdap-stream library (https://github.com/demlabsinc/libdap-stream) and consists of several levels.

The first level is the title of the package itself

Stream packet header

8 bytes : Flow packet signature

1 byte : Stream packet type

4 bytes : Packet size of the stream

8 bytes : Sender's address

8 bytes : Destination address

Next after the header is the body of the stream packet, encrypted by the stream key, received through the session protocol. Inside the body of the flow packet, there may be a plurality of so-called channel packets. Channel packets lie inside the encrypted part, so by default they consist entirely of unencrypted data, although in the case of VPN there may appear one more level of encryption, the third one on account (1st - session, 2nd - streamed and 3 - channel).

Channel packet header

1 byte :channel packet type

1 byte : Type of channel encryption (0 if not present)

1 byte : Data type subtype (binary, command, information, audio, video, network, etc.)

1 byte : Padding (not used)

8 bytes : Serial number. Used to control the order of the data, if not then 0

Channel packet's body

Actually the body of channel packet. It can include a block, a request for a transaction, a piece of network data for VPN, an audio video frame for streaming, or just some abstract piece of data sent back and forth for a modest fee

3.6. Services

The DapCash plug-in is universal, accordingly, the DapCash network can provide various services. At the start, it will be smart contracts and VPN, in due course the distributed database, file storage, videoconferences, etc. will be added.

4. Additional Features

Next come the enumeration of the planned and projected additional features of the DapCash project.

4.1. Smart contracts

Standard smart contracts on Solidity and other languages compiling their results in EVM byte code. EVM will be fully supported, including the ERC20 token and its own standard. Signatures for the code of their own distributed DVM virtual machine and, accordingly, smart contracts for it are also reserved in advance.

4.2. VPN service

Through a session and streaming DAP protocol, it is logical to present a VPN service on the same nodes of the network, provided that they will of course include the appropriate configuration. VPN service will be provided through the exchange, where different sellers will expose their prices for the time using and / or the consumed traffic. Further the user can manually or by means of automatic algorithms choose to himself one or several service providers, pay them through DapCash and use it further. "Onion" routing between nodes and anonymous cash transactions will make it possible to create completely anonymous network connections while maintaining high quality of communication.
This should eventually give us a decentralized VPN with high quality of communication and minimum prices.