I/O Digital Currency

General purpose hierarchical Decentralized key-value storage on blockchain, Private DIONS

I/O Coin Development team

November 2nd, 2014

1. Introduction

The development of modern cryptocurrencies began in 2008 with the publication of an article by Satoshi Nakamoto [1] and Bitcoin release, although it should be noted, some work in this direction was made earlier[2]. In Bitcoin resistance against rewriting (also known as double-spend) is supported by the mechanism of Proof of Work based on the double-sha256, which requires from attacker to have more computing power than other "honest" miners altogether. Some solutions marked by Bitcoin development team were criticized by experts and as a result alternative currencies were created. NameCoin suggested to use blockchain as a distributed database. For the first time Namecoin implemented merged mining method, which allows to protect the namecoin chain with Bitcoin network[3]. Other researchers have proposed changes to the algorithm PoW, this as it was thought to be more resistant against centralization, for example, Litecoin[4] (scrypt), Primecoin[5] (searching for prime numbers), DarkCoin[6] (a combination of 11 hash algorithms). Even alternative mechanisms of block generation, such as PoSfamily - PoS[7] (PeerCoin), PoS2.0[8] (BlackCoin), PoS I/O[9] (IOCoin), PoS-V[10] (Reddcoin), DPoS[11] (Bitshares), as well as PoB[12] (proof of burn). Coins specializing on information storage Datacoin[13]. There are various alternative currencies built on the similar principles Ripple[14], Nxt[16], Qora[15]. In 2013-2014, a new breed of coins came about, offering anonymized transactions based on different algorithms, such as centralized mixing[17] FedoraCoin, decentralized (with masternodes) mixing[6] Darkcoin, ring-signature[18] Bytecoin. They proposed a shift from public addresses to the stealth-addresses, for the preservation of privacy (hiding different transactions made to the same address from each other) [19]. Also in 2013-2014, the ethereum project was born. Ethereum proposed the possibility for any developer to build and publish next-generation distributed applications in a decentralized network. [20].

Observation of cryptocurrencies shows that coins that fail to implement new technologies die quickly (except coins with real-world adoption, like Bitcoin, Litecoin, Dogecoin), and hence after the release of ethereum we will wait for "Mesozoic extinction," with the appearance of new, ethereum-based currencies.

2. I/O Coin Private DIONS Proposal

In this paper, we propose a new language of transactions I/Oscript, which is the development of the script, allowing construction on the basis of our decentralized blockchain for general purpose key-value storage with an advanced system for delegation of rights, as well as further steps to develop I/Oscript to Turing-complete language, which will grant the same ability as ethereum, to create smart-contracts. The article will be organized as follows. In the second section, we will discuss the different methods of using centralized key-value storage and discuss what properties it must satisfy for the most-wide use. In the third section, we will propose new operators expanding script to I/Oscript. In the fourth section, we briefly outline the next steps in the development of I/Oscript.

3. Usage of Decentralised storage

Decentralized key-value storage can be used in various ways. One of those ways is the aliases, (decentralized implementation of IONS) as enshrined in blockchain matching aliases to-> addresses. An improvement of this solution will be the addition of an extra message field to all transactions, as well as the expansion of aliases on Private-Addresses. This will allow store/exchange/application. To put this into a real life perspective, it would mean that you will be able to use the brand name, track purchases from different buyers, and at the same keep their balances private.

4. Real Life Application - Proposal

Now let us put this into action to further identify its applications. Imagine if you will, that a local pizzeria wanted to implement this platform. Well, firstly they would be assigned an alias such as perfect_pizza-io to replace the existing address such as "iXBAV3Cf48rvv7DEmwiqqHXXTDvTGwJnSk". Now if we were to place an order through this pizzeria we would use the alias provided and to further distinguish between orders from each other they provide you an order id, so they ask you to pay a certain amount at pizza_next_-door-io:77126. Your wallet will match alias pizza_next_door-io, understand that this is private-address and generate a unique private address. Next step is to send to this unique address a transaction adding "77126" to the message field.

5. Real Life Application - Limitations?

There are however some issues - such as cybersquatting and typosquatting - with using such a system in a decentralized manner that other platforms such as NameCoin have also faced. To overcome such an obstacle we are proposing the use of an arbitrator to resolve disputes. The problem with this solution is that it goes against the core principles of Bitcoin to provide a fully decentralized environment. It also creates a point of failure whilst harming the integrity of the network as it can be manipulated by those who gain control of the arbitrator. For this reason we cannot trust a centralized arbitrator to control and manipulate the outcome. This is especially true when we are dealing with assets as they could potentially be open to intrusions and unwanted control. Let us consider an asset such as gold and place it into a stop-loss contract which would sell a given amount of this asset based on its live market price. How do we trust that the system provides the prices with a high level of integrity and on top of that, how do we insure that our assets that are in such a contract are safe from being sold without a true market change but rather manipulation of the index controlling the contract?

For example, you want to create a stop-loss contract, ie to sell a certain amount of goods (eg crypto Asset gold) if its price falls below $ 1,000 per ounce. You trust the information provider Bob and Kelvin, who regularly updates the information in the stored decentralized field "bob_gold_price" and "kelvin_gold_price" respectively. You are posting a transaction (using the pseudo code for understandability).

For various reasons, you can not trust a centralized arbiter and therefore do not want it to be able to change (or transfer the right to change) to someone else. In addition to the possibility of using variables in the transaction, to reduce the size of popular transactions, there must be the ability to store the code of transactions in the storage. It is necessary to to forbid all changes in these variables, otherwise the contract that you signed when you create it can change entirely at the time of execution. In addition, there are many uses of decentralized storage, such as proof of time, storing private zerocoin-containers, the use of 3rd-party data (eg maintain compliance the user -> pgp-key in a decentralized storage).

6. Real Life Application - Solutions!

To overcome such limitations and progress towards a fully decentralized storage method, I/O has proposed the use of two new op-codes: **OP_REGISTER** and **OP_UPDATE**. The first **OP_REGISTER** will correspond to 6 elements, alias, value, type, height, pubkey, sign. alias - up to 256 characters string composed of characters "a-z" case insensitive ('A' is the same as 'a'), digits 0-9 and the characters "_%.", where '.' is a special symbol. A lower limit on the length of the string depends on the "type" and can influence the requested fee value - an arbitrary string (length limit individually for each type) type - belongs to enumeration admissible at the moment types. The first will be implemented type "alias" height - number of blocks that will be reserved for the record. Restrictions on height is individual for each record type. For example, to type equal "alias" and type equal "code" independently from parameter height reservation will be infinite. pubkey - public key corresponding to the private key which control this record. sign - the signature of the previous five parameters (not the entire transaction) with a secret key. Signature depends on the alias. If the alias does not contain '.', the sign should correspond to the public key. If the alias composite, such as "local_department.specified_ direction.big_company_name", a sign must match the key of any parent alias, such as "specified_direction.big_company_name" or "big_company_name", of course for declaration alias "a.b.c" must be registered alias "b.c" and "c". For the record type "alias" exists supreme pubkey which belongs to the developer, so the developer, as the referee is able to update all the records of type "alias". At the same time for other types of records "dev's pubkey" is not the parent, so other types of records cannot be altered by anyone other than their owners (and for some types of records changes cannot be made at all). **OP_UPDATE** will pops from stack alias, value, type, height, newpubkey, newsign, old sign where the parameters correspond to those in **OP_REGISTER**. New pubkey may coincide with the previous, new sign signs first 5 parameters, oldsign sign 6 parameters with the private key corresponding to previous pubkey.

Record type specifies the corresponding associated fees, so unlike bitcoin-script the code of the transaction and its validity affected by the amount of coins in a transaction. So, to create a record type "alias", someone will need to transfer a certain amount of coins to the developer (otherwise the transaction will be invalid), in this case it will be a fee for maintenance aliases system. At the same time for other types of records will also be required to pay, but the corresponding coins will be destroyed. For certain types of records will be provided a high registration fee, most of which will be returned when you remove it from the blockchain (you can delete the record, by using **OP_UPDATE** with height set to 0). This mechanism allows to remove from the base of unnecessary information and to reduce the load.

7. Registration Fees

The record types specify the fees involved with its registration. If you were to register an alias, the coins used would verify the address as being yours and its fees will be kept for the maintenance of the alias system. In other cases the coins sent will be destroyed after verification. There are some records that carry a high registration fee due to the nature of the contract/record. In these cases, most of the coins will be returned once the contract has been removed from the blockchain by using "OP_UPDATE with height set to 0". This removes any excess bloating of the block chain and reduces the overall load on the network and its users.

8. Future Development

When we were considering the future direction of this development, we wanted to leave room for storage of records in a manner that the fees could be modified and also to prevent D/DoS on the nodes.

Other opcodes such as **OP_EVAL** and **OP_LOAD** will be introduced to allow the developers to revise the conditions of the fees so that they can be calculated from both the size of the transaction and also the number of steps involved with its procedure. The latter is the method used to prevent DOS attacks and bring the I/O Coin platform to the same level to that of Etherium.

9. References

[1] Satoshi Nakomoto https://bitcoin.org/bitcoin.pdf

[2] Hal Finney http://cryptome.org/rpow.htm

[3] Vincent Durham https://wiki.namecoin.info

[4] Charles Lee https://litecoin.info/

[5] Sunny King http://primecoin.io/bin/primecoin-paper.pdf

[6] Evan Duffield, Kyle Hagan https://www.darkcoin.io/wp-content/uploads/

2014/09/DarkcoinWhitepaper.pdf

[7] Sunny King, Scott Nadal http://peercoin.net/assets/paper/peercoin-paper.pdf

[8] Pavel Vasin http://www.blackcoin.co/blackcoin-pos-protocol-v2-whitepaper.pdf

[9] http://www.docdroid.net/hvjr/io-coin-pos-io.pdf.html

[10] Larry Ren https://www.reddcoin.com/papers/PoSV.pdf

[11] Daniel Larimer http://bitshares.org/delegated-proof-of-stake/

[12] P4Titan https://slimcoin.org/images/downloads/slimcoin_whitepaper.pdf

[13] http://datacoin.info

[14] https://ripple.com

[15] http://qora.co.in

[16] http://www.nxtcommunity.org/nxt-whitepaper

[17] http://fedoraco.in/

[18] Nicolas van Saberhagen https://cryptonote.org/whitepaper.pdf

[19] http://sx.dyne.org/stealth.html

[20] https://www.ethereum.org/pdfs/EthereumWhitePaper.pdf3

10. I/O Coin Development Team

Whitepaper: Max Antonenko

Lead Developer: Derek Hatton

Project Manager/ Developer: Joel Bosh

C++, RPC Lead Dev: St. Patty, OPhie

Mobile Developer: Unek

Community & Marketing Manager: Richard Groen

POS Security & Systems Management: Sam

Project Graphic Design: crz

Video Graphics: Michael Jonsson

Wallet Compiling Dev: Wuher

Wallet Compiling Dev: pcmerc

Editor: Evok3d

OP Beta Testing: Valianthor

Code Reviews: Rat4

I/O Digital Currency