Randpay: The technology for blockchain micropayments and transactions which require recipient’s consent

Oleksii Konashevych, Oleg Khovayko

Erasmus Mundus Joint International Doctoral Fellow in Law, Science and Technology (Consortium LAST-JD, EU)
Chief Technical Officer of Emercoin, USA

ARTICLE INFO

Article history:
Received 22 July 2019
Revised 2 April 2020
Accepted 18 May 2020
Available online 26 May 2020

Keywords:
Cryptocurrency
Electronic Lottery Tickets
Blockchain
Emercoin
Micropayments
Randpay

ABSTRACT

Randpay is a technology developed in Emercoin for blockchain micropayments that can be more effective in some scenarios than the Lightning Network, as seen in the paper. The protocol is based on the concept of Ronald L. Rivest published in the paper 'Electronic Lottery Tickets as Micropayments' (1997). The "lottery ticket" was designed for centralized systems where a trusted third party is required to provide payments, and in some scenarios, is also a lottery facilitator. The existing blockchain protocol cannot accommodate peer-to-peer "lottery" micropayments at least, without creating payment channels. Therefore, the implementation requires the development of an upgrade to the blockchain core. In the result, RandpayUTXO was introduced – an infinitely spendable zero output that requires the payee's signature to be published in the blockchain. Randpay is considered to be the first blockchain protocol to require the payee to sign the transaction by their private key. This is a significant feature to improve, not only for transactions but also extend the use of the blockchain for legal deeds that require a payee's consent to be recognized in legal applications. The second important innovation of this research is the implementation of Blum's 'coin flipping by telephone' problem to design a 'lottery ticket' that does not require any third party to facilitate the lottery. The paper offers an analysis of the mathematical model, and proof of how 'lottery' can be beneficial, an API description is also added. There is also an attack analysis and an overview of existing solutions.

© 2020 Elsevier Ltd. All rights reserved.

1. Introduction

The problem of micropayments on the blockchain (Nakamoto, 2008) is that they trigger several issues in the scalability of the system: blockchain bloat, bandwidth, growth of fees, issues of trust, and security.

The Lightning Network (Poon and Dryja, 2016) is a well-known project designed for micropayments. Such projects as SegWit (Understanding 2018) (reduction of included data in the ledger), Ethereum’s sharding plan (Sharding 2019) (segmentation of the ledger), and Ardor’s two-level (parent-child) blockchains (Ardor 2019) are aimed at addressing some of the scalability aspects. However, they do not address the issue of micropayments and should be considered instead as complementary solutions for micropayments than an alternative.

The problem of system load and ledger bloat due to the high amount of records produced by micropayments is not an exclusive blockchain issue. It also relates to earlier centralized technologies, where the most common approach to address these issues is an aggregation, which is supported by the following conclusions. Payment aggregation replaces many micropayments with a small number of total payments to be recorded in the ledger. With the aggregate, transactional payments (fees) are paid only for such consummated transactions. In other words, aggregation reduces not only the number of entries but also the transactional costs per payment. There are two types of aggregation in centralized systems: 1) accurate; for instance, all phone calls are accounted but paid as a lump sum once a month, and 2) probabilistic. One of the most known probabilistic protocol was proposed by Ronald L. Rivest in 1997 when he published his research on 'Electronic Lottery Tickets as Micropayments' (Rivest, 1997).

There are no known mass implementations of Rivest’s method, probably because in central-server systems, there are other more effective approaches.

The implementation of Rivest's lottery ticket required redesigning the standard blockchain Emercoin protocol, as well as the Rivest's protocol itself, because it was designed for three parties: seller, buyer, and bank (broker).
Emercoin was chosen as a development stand, and it has the following features. Launched in 2014, it is a combination of Bitcoin’s PoW and Peercoin’s PoS (initially was adopted the original Peercoin PoS, but then, the security was improved) currently running with the approximate ratio 1:6 of blocks created by PoW and PoS respectively; hash rate is approximately ¼ of Bitcoin because of the merged mining protocol that allows nodes simultaneous mining of both Bitcoin and Emercoin. Randpay concept was implemented in the wallet from version 0.7.1 (Emercoin 2017).

The off-chain portion of Randpay was challenged by the requirement to exclude trusted third parties; to provide users to interact with each other peer-to-peer, and at the same time, not to use an existing approach for peer-to-peer protocols that require the creation of so-called “payment channels” because they typically require also performing opening and closing blockchain transactions (but the aim was to reduce them).

The solution was found based on Blum’s ‘coin flipping by telephone’ problem. In the findings, the Randpay protocol works off-chain and does not require third parties to facilitate the lottery play-act, and only payable transactions are directed to the blockchain. At the same time, there are no “channels” as found in other protocols. Therefore, anyone can pay to anyone and settle the transaction without opening the channel, maintaining the state of the channel, and closing it to pay or to release the funds back.

The core idea of Rivest’s ‘lottery tickets’ is to aggregate micro-transactions using a probabilistic method. During the purchase, the parties settle the transaction not with the payment, but with a ‘ticket,’ which is winning for one of two sides: either the buyer or the merchant. The probability on the side of buyers who will not pay if they win this ‘lottery.’ If the merchant wins, the buyer pays, and in this case, only this transaction is recorded to the blockchain ledger.

Presumably, the ‘lottery ticket microtransactions’ are ideal for regular small payments; for example, phone calls with per-second charge. The merchant regulates the price and the probability in a way so to receive payments fewer times but in more considerable amounts.

Buyers with a higher probability do not pay, but when they do pay, the price is “probability times” higher than if it was a regular payment. Nevertheless, they should not feel like they are being mistreated, because they did not pay for services in previous ‘plays’ and with the same probability will not pay in the future. In the long term, paid and unpaid services tend to equate to a fair balance for both sides, which is proven, using probability theory discussed later. Of course, the payment amount in the ticket and the probability all are a matter of bargain; any of the parties are free not to accept an unfavorable deal.

2. Micropayments retrospective

This section outlines the chronology of academic discussion on micropayments, which takes place over three decades. As it comes from the analysis, most of the protocols are not relevant for comparison with Randpay because they require third parties and are designed for centralized payment systems. As to found protocols for micropayments in cryptocurrency, they either require payment channels or are not flexible in terms of probability variation.

2.1. Pre-blockchain micropayments

The issue of the effectiveness of electronic payments was raised with the appearance of electronic payments themselves. The academic interest to micropayments began in the 90s of the twentieth century with multiple issues on the agenda: the computational power of machines was not enough, the cryptographic schemes were not so developed, and online payments were expansive.

During decades of academic work in micropayments appeared a few noticeable overviews, surveys, and evaluations: “The Nuts and Bolts of Micropayments: a Survey” (Ali et al., 2017), “Micropayments overview” on W3C web-site (Micropayments 2019), a chapter “Micropayments” in Kou’s book on “Payment Technologies for E-Commerce” (Kou, 2003), “Evaluation of Micropayment Transaction Costs” (Papaefstathiou and Manifavas, 2004), “Comparing and Contrasting Micro-payment Models for Content Sharing in P2P Networks” (Dai et al., 2007), which can be relevant for an in-depth study of micropayments.

In the analysis, it is found two different directions of the research in the pre-blockchain period: (1) how to increase the bandwidth of electronic payments and so to make possible small payments, and (2) probabilistic methods of aggregation of payments, which were meant to reduce the number of actual payments, and thus to enable micropayments without the need to significantly increase the performance of the electronic system and the Internet.

In 1997 Ronald Rivest proposed the protocol for probabilistic payments, which he called the “lottery ticket.” (Rivest, 1997): “The probabilistic nature of lottery tickets makes payment of small values simple. For example, an electronic lottery ticket for a $10.00 prize with a 1/1000 chance of winning has an expected value of one cent. A user can pay a vendor one cent by giving the vendor such a lottery ticket.”

2.2. Blockchain micropayments

With the appearance of blockchain technologies due to its limited bandwidth, the issue of microtransactions raised again. Pre-blockchain works are not relevant for further review because they are designed for centralized systems and necessarily require at least one (but sometimes more) trusted third party: a bank, a broker, or in some schemes, a “lottery” facilitator. Pass and Shelat (Pass and Shelat, 2015) proposed their protocol for cryptocurrency based on Rivest’s lottery. Researchers introduced two basic approaches: the first requires changes in the blockchain protocol to enable peer-to-peer lottery tickets between users. The solution differs from Randpay as it is non-flexible in terms of choice of probability. The authors propose x100 by default. It is supposed that any other probability rate to introduce in the protocol requires the upgrade of the blockchain protocol. Therefore, to satisfy the multiple needs of a business, there must be deployed a wide range of “probabilities” for users’ choice. Moreover, the proposed method allows the implementation of only even figures because of the binary nature of the scheme. Thus, if the seller needs the probability of x3…x6…x1003…etc., this system does suit them.

The second proposed method consists of two variations: with an escrow and with an “invisible” escrow, which is not relevant to compare with no-third-party Randpay protocol.

As to the effectiveness of the Pass and Shelat proposition, the protocol requires at least one on-chain payment transaction to lock the possible future payment and then to play the lottery with this payment, and then when the seller wins, they perform the transaction to send this amount to the seller.

Say, the counterparties play the lottery with the token of 0,0001 BTC. The buyer first needs to create this amount by sending this amount from the available balance in the wallet to the escrow address. Then they play, and the token will not necessarily be paid with 1:100 probability, i.e., the buyer has 99 chances to get the product for free, and one chance that the seller will get this amount which equals to the cost of 100 products, the seller’s winning transaction will be performed off-chain.

The scalability of this scheme is questionable. The first issue here is that the customer needs to have a lot of payable tokens of a specified amount. In our example, many times by 0,0001 BTC. Moreover, for different services, it might be necessary to have a lot
of tokens of a different amount, the same as if the user had to pay by 25-cent coins for road tolls, and 5-cent coins for calls from public phone booths, they need to have a handful of relevant coins. So, to have these tokens, the user will need to perform a transaction(s) of splitting the lump sum, which adds load on the ledger.

The second issue appears from the continuous nature of some services where the client pays while consuming these services and may run out at a certain point. Say, the client makes a call and has to spend 1 Satoshi to buy additional time during the session to continue talking. If the client has not prepared enough Satoshis, they will not be able to pay, and the call will be interrupted.

Therefore, clients must prepare beforehand enough payable coin tokens, which may require a relevant number of blockchain transactions, even if they will not be paid after because of the probabilistic nature of such a payment scheme. In case the token becomes payable in the lottery, it requires closing the transaction, which leads to two total transactions per such micropayment.

There is no information on the mass adoption of this scheme. As it comes from the paper, the protocol has been prototyped on the testnet. Hu and Zhang (2018) proposed to improve Pass and Shelat protocol by setting up a time-locked deposit, whose secure utilization is assured by the security of a primitive called accountable assertions under the discrete logarithm assumption, their scheme reduces the number of on-chain transactions to one, and yet maintains the original scheme’s advantages. However, authors notice that as long as both sides of payments are honest, their scheme can be conducted without any third party’s involvement and require at most one “on-chain” transaction during each execution. Among non-probabilistic schemes for cryptocurrency micropayments here to be mentioned: DAM scheme (Chiesa et al., 2017), Micropaying to a Distributed Payee with Instant Confirmation (Ni et al., 2018), Orchid (Salamon et al., 2018), Efficient Micropayment of Cryptocurrency from Blockchains (Rezaeibagha and Mu, 2019), SLIP (Zhong et al., 2019) and Deploying PayWord on Ethereum (Elsheikh et al., 2019). Mentioned papers discuss different aspects of the off-chain payments using payment channels.

Recently launched The Lightning Network (2016) (Lightning Network 2017) is an off-chain protocol that can handle micropayments (non-probabilistic method). A more detailed discussion on the problem of off-chain payment channels is provided in Section VI: The Lightning Network Comparison.

3. Randpay concept

In this section, it is explained how Randpay protocol was designed. In the first subsection, it is discussed the concept of probabilistic payments. The second subsection explains the probabilistic nature of profits in lottery ticket payments in comparison to classical payments. The next subsection shows how the off-chain part of the protocol provides for peer-to-peer interaction of parties, where they play the lottery. In the beginning, it is explained what objectives were defined, and which issues were to be addressed by this part of the protocol. The last subsection explains how the random choice of the payer becomes an on-chain payment when the payee wins the lottery and how malicious behavior is addressed.

3.1. Understanding 'lottery tickets as micropayments’

Randpay can be used to sell any items; however, it is best suited for virtual goods and services that cannot be purchased traditionally by a direct personal exchange of cash because of the gap at a moment or place of exchange.

Randpay makes sense for remote mass distribution of products, which are paid as they are received at a tariff: for example, phone calls with per-second charging, road tolls with per mile payments, sharing information (market tickets, news), sharing media, etc.

A hypothetical client, let us call her Alice, is the recipient of goods or services for which she pays cryptocurrency. A merchant, Bob, sells these goods or services for which he expects the payment.

The essence of the idea is to finalize each settlement, not with a payment but with a ‘lottery ticket.’ Only Bob’s winning ‘lottery tickets’ will be published into the blockchain as the transaction.

Bob provides Alice a “lottery ticket,” which carries the information of the space of payment addresses, where one is Bob’s winning. Alice makes her random choice picking one address from the provided space, generates the raw transaction, and sends it directly to Bob. If Alice’s choice contains the payment address to which Bob has the private key, he will sign the raw transaction and publish it on the blockchain and so he will take the money.

If Alice has chosen a payment address to which Bob does not have a key, this transaction will not be published and just set aside, and Bob will deliver Alice the product for free.

As evident from this scheme, no opening transaction is required, counterparties play the lottery off-chain peer-to-peer, and only a payable ticket is published on-chain.

3.2. The mathematical expectation of profit vs. classical payments

The mathematical expectation of the transferred amount for each lottery ticket is equal to the amount that must be received in a single payment act.

For instance, the provider Bob receives 2 cents for each call per minute using classical payments. From 5 communication sessions supplied, Bob will receive 10 cents (2 * 5 = 10). But if he uses a ten-cent ‘lottery ticket’ with the probability of winning 1 to 5, the expected value of five deals will be the same 10 cents. He will not necessarily receive this money in the result of 5 transactions, because the payment is random, but in the long run, he will be receiving payments with the expectation of 1/5 (20%) and his financial result is calculated across all payments from all clients (if Alice did not pay, Charles would have the same probability of 1/5, and the same in other clients). The mathematical proof of expected linear income is seen in Section V. Economic Analysis and Mathematical Model.

The client plays the so-called Russian roulette, where for the mentioned example, there are five sockets in the revolver, and only one is charged. When she lands on it, Alice will pay 10 cents (instead of 2). The average rate of five purchases is the same as the initial 2 cents (10/5 = 2). See Fig. 1.

However, such a distribution of risk/benefit may be disadvantageous. Therefore, counterparties can customize the amount of the ticket and the probability of payment.

In the same example, but with a probability of winning 1 to 100, only every hundredth sale according to the mathematical expectation will be paid. For Bob not to lose compared with classic payments, the ticket price must be 200, i.e., the average tariff for one hundred transactions also tends to become the same 2 cents (200/100 = 2).

Thus, with higher probability and thus, higher frequency Alice gets an entirely free session, or with a smaller probability for 1 minute, she overpays a lump sum equal to x (probability) times cost of the price per product.

For Bob, the value of payments also tends to be a fair amount, which would be obtained with multiple realtime payment acts and what is essential, not necessarily from the same buyer.

At the same time, the number of actual payments of cryptocurrency also decreases by a factor of 100 (in the last example), and the transaction costs (fees) are reduced proportionally.
The problem with the peer-to-peer protocol is that if Alice is to guess the number of Bob’s fingers behind his back, Bob can manipulate, showing after Alice’s choice any non-winning result.

Blum offered the solution in his ‘coin flipping by telephone’ problem (Blum, 1983). Alice and Bob want to flip a coin by phone, but they are located at a distance from each other and can only communicate via a communication channel.

Bob selects a random sequence of bits b, writes it on a piece of paper, locks this sheet in the drawer, leaves the lock key, and sends the box to Alice. It is assumed that, without having a key, Alice cannot get to the contents of the box. After receiving the box, Alice chooses a random bit c and sends it to Bob. In response, Bob sends Alice the key to the box. The outcome of a coin flip is the d = b ⊕ c. The issue of the coin flip is solved with the help of one-way functions, where input messages cannot be retrieved from the hash sum (checksum, digest) of it.

Randpay protocol is based on Blum’s concept, and offers an original implementation considering that it was designed for blockchain transactions.

Besides, hashing is a one-way function. There is another crucial feature attributed to strong cryptographic algorithms that underlay the found solution. It is barely possible to find logically close outputs, for example, consecutive numbers (‘1, 2, 3 …115’). To find a logical sequence of hashes, the attacker will need extremely high computational power irrelevant to the value of the transaction.

The algorithm of generating a lottery choice range can be explained in this simplified mathematics. For instance, Bob decided to use number 115. He must divide it by the number of choices in the lottery. Let us call it a ‘risk.’ For instance, 10, that is 10% probability chances for Alice to guess it.

Then Bob cuts off the decimal and keeps it secretly from Alice. Therefore, he obtains:

\[ 115 / 10 = 11.5 \]
\[ 11.5 - > 11 \] (1)

Bob sends the number 11 to Alice (let us call it ‘CHAP root’), specifying that the probability is 1 of 10 (ten addresses to choose from where one is winning).

Alice multiplies 11 by 10 and gets = 110, which is the base for guessing.

To play this lottery, she adds an arbitrary number of the array 0-9.

Let us say Alice chooses 3 and creates the transaction for Bob with the address retrieved from 113 (110+3). Since Bob’s private key is 115, Alice’s choice is useless for him; he lost this lottery. Alice does not pay and receives the product for free.

In case Bob lost, he can deny supplying the product, so Alice loses nothing: she neither pays nor receives the product for free.

The mentioned operations in this simplified example are performed indeed with something that is called the “raw address.”

A set of operations generates the blockchain (cryptocurrency) address. First, the merchant Bob creates an asymmetric pair (public and private key), then from the public key, the system generates a digest using SHA-256, which then becomes an input to retrieve a hash using RIPEMD-160 algorithm. Then the system performs another set of operations that are required to retrieve the blockchain address, which is shown to the end-user, see (Bitcoin address) for details.

RIPEMD-160 checksum is a ‘secret’ of the lottery and the initial figure from which the system generates a space of addresses, mentioned in the example as ‘CHAP root.’

Therefore, the businessman Bob offers Alice to choose to generate one payment address from the range of available choices [110,119], where only one has a private key, so Bob will be able to dispose of it.
Alice chooses one address from the proposed range and then creates a normal payment transaction on the blockchain to this address, but this transaction she sends not to the blockchain but directly to Bob, so he can verify it and publish it himself. Bob will see if the payment amount is correct or not and check if he has a private key to the payment address; if not, then Bob lost this lottery.

“Addrcap” is the lottery ticket with the space of addresses to pay. The buyer makes a choice and packs the raw transaction, which sends directly to the buyer. In the next subsection, it is discussed how the buyer verifies and signs it to publish on-chain.

3.4. Naïve ‘lottery’ microtransactions on the blockchain, or why RandpayUTXO was developed

The typical blockchain protocol, for example, Bitcoin, as well as similar Emercoin, at first glance, have the necessary set of payment scripts to create a resulting lottery transaction.

However, businessman Bob may behave non-cooperatively and publish the transaction, regardless of whether the payer Alice chose Bob’s winning address or not.

If Bob publishes the transaction, which is a non-winning for him, he will not be able to access the money because neither he nor others have the private key to spend this money in the future. Alice’s money would be considered lost forever.

To protect Alice from Bob’s malicious behavior, a proof-of-ownership of address ‘RandpayUTXO’ was developed, purposefully to prevent a transaction in which Bob does not have the proper key.

A cryptocurrency transaction consists of two main groups of elements:

1) input(s) that is at least one address from which the money is to be spent (“spending”); and
2) output(s) is at least one address to record the spending, and other address(es) if necessary, to record a change.

The input of a new transaction must necessarily be the output (UTXO) of a transaction recorded somewhere in the previous blocks or the mempool. The only exceptions to this rule are transactions that refer to the generic block or cryptocurrency, which was just created (“mined”).

In the result of the development of the Randpay protocol, in outputs, so-called UTXO set, there was planted a special unspendable dummy output number 0 from a non-existent transaction with the following ID:

\[
\text{TXID} = \text{EC}\text{EC}\text{E}\text{E}\text{CE}\text{CE}\text{C}\text{E}\text{C}\text{E}\text{C}\text{ECE}\text{C}\text{E}\text{EC}\text{EE}\text{C}\text{E}\text{C}\text{E}\text{CE}\text{CE}\text{E}\text{C}\text{E}\text{C}\text{E}\text{CE}\text{C}\text{E}\text{C}\text{E}\text{CE}\text{C}\text{E}\text{C}\text{E}\text{C}\text{E}\text{C}\text{E}
\]

This special output, called RandpayUTXO, is included in the transaction with the Input and must be signed by Bob’s private key but not Alice’s.

The network will not accept a transaction in which at least one spending (Input) is not signed by the private key of the address at which it is recorded.

The original blockchain protocol does not require any action from a recipient, which means a transaction can be sent without the payee’s consent or even knowledge of it. However, with RandpayUTXO, the transaction needs to be signed by both parties.

RandpayUTXO has the following properties:

- it can be repeated, and endlessly spent, that is, after the transaction, it is not marked as spent;
- user can only spend 0 coins;
- this UTXO can only be present once in a particular transaction as input, specified as ‘randpay-in’ here;
- the randpay-in must be signed by the private key of the current output (receiving) ‘vout[0]’ address.

Other words, the output address ‘vout[0]’ is copied into the input ‘vin[0]’ during validation and, as a result, the ‘vin[0]’ must be signed by the recipient’s private key, associated with ‘vout[0]’, to be accepted. See a conceptual scheme in Fig. 2.

When Alice has chosen one address from the address space offered by Bob, she creates a draft blockchain transaction, where:

- in the input array she adds the dummy address RandpayUTXO;
- in the output ‘vout[0]’, she puts a chosen address from the lottery.

Then Alice signs all inputs except RandpayUTXO and sends it off-chain (directly) to Bob. Bob signs dummy input if Alice has chosen the address to which he has the key and sends the transaction to the blockchain. That is, how Bob’s winning transaction becomes blockchained, but in the case of a loss, it will be dropped off.

The improvement of the blockchain resulted in the first-ever implemented protocol that technically requires a recipient’s private key.

It should be noted, however, that this invention has another significance beyond the protocol for microtransactions. There are some types of legal transactions where the recipient’s consent is required because otherwise, it does not acquire legal force. For example, in some jurisdictions, in a deed of gift, the recipient must accept the gift. It also can be applied to transactions that require consent from authorities, which are not at the same time beneficiaries. For example, the sale of a land plot may require consent from the government. In this case, at least two addresses will be specified as output: the one who receives an asset and the other which gives the consent using RandpayUTXO but receives nothing or a part of the payment (for example, as a fee “duty stamp”).

Therefore, RandpayUTXO can be applied not only for micropayments but also for other payments to ensure the payee’s consent. Emercoin RandpayUTXO introduced the technology which makes blockchain transactions closer to the real world with a variety of legal constructions that users may require.

3.5. Tests

This subsection presents the results of laboratory tests and methodology.

To perform the tests, a script which emulated interaction of the parties (payer&payee) was written, with the following settings. The “risk” - 1/3, the payment - 0.5 EMC. Therefore, the average rate of the product is 0.5/3 = 0.166 (EMC).
Testing resulted in Randpay acts, which consisted of 7 off-chain lottery plays and two on-chain recordings when the script randomly picked payable (winning) address.

For this research, the script ran seven times. Two winning ticket were obtained. The raw transaction unpacked and verified successfully, signed, and published in the testnet. The mathematical expectation of the payment for ‘0.5 EMC ticket’ per transaction was: $0.5/3 = 0.167$ (EMC). The actual result was $1/7 = 0.142$ (EMC).

The code of the script, transactions, and links to the testnet are presented in Appendix A, “Randpay Testing.”

Tests are performed successfully. Results showed that the technology works, and users can achieve the expected “probability/risk” for their payments.

4. Attack analysis

Randpay is a peer-to-peer protocol that is designed to prevent malicious behavior and does not require any third party for settlement.

As a result of a lottery, the client Alice can lose and therefore pay a lump sum, which is larger “probability” times than if it was a classical purchase.

Bob’s safe strategy is to supply the product not before but after the ‘lottery’ action. Therefore, in the worst scenario, Alice does not pay the lump sum but does not get the product – nobody loses.

There is, of course, the possibility that Bob will receive money but will not supply the product. But this is a general problem attributed to both conventional payments and Randpay and stands beyond the purpose of this research. Alice’s strategy to minimize her risks here as a client is not to agree to a deal with an unacceptable high payment or use an escrow (third party).

Among the general security issues is authentication. After the step of choosing the product on the website, Alice goes to the step of Randpay play and payment in her wallet, which is a different system. Alice must be confident that she continued interaction in the wallet with Bob, as well as Bob knows that this is Alice sending the transaction. There must be an authentication mechanism in place, presumably on the merchant’s (Bob’s) e-commerce/website platform. Though more likely, the merchant Bob will ask Alice to specify her cryptocurrency address in the website account. Nevertheless, there are other ways to authenticate a client/merchant, and the Randpay protocol does not limit how the authentication process on the platform is designed. In all cases, parties will have to stick to the native blockchain’s mechanism of transaction authentication through public-key cryptography.

In the previous section, it is explained “addrchap” and “RandpayUTXO,” which excludes attacks on the protocol from Bob. Let us consider attacks from the side of the client Alice to the Randpay system and measures to prevent them.

Attacks on the Randpay subsystem can be divided into two groups:

– Alice creates incorrect transactions to pay less or not pay at all,
– Alice cheats Bob by double spending (‘double spend’ is referred to as ‘DS’).

To perform the attack analysis, it is researched all elements of the system. Thus, the description of the API and URI protocol is provided in Appendix B and C.

Briefly, the protocol consists of ‘WANTPAY,’ which is Alice’s request to Bob, where she specifies the amount of payment and probability. This step is not obligatory; presumably, it is for bargaining. The second step is obligatory ‘NEEDPAY’ where Bob sends the request (or response to the previous step) to Alice, where he sends the lottery ticket ‘addrchap’ (space of addresses to pay), payment amount, and probability. The third step: Alice makes her random choice from ‘addrchap’ and sends it to Bob. Bob’s wallet verifies Alice’s message and sends it to the blockchain if Bob has the private key to the chosen address.

4.1. Attacks on the transaction

The attack scenarios discussed in this subsection are impossible in Randpay because the current code contains mechanisms to prevent them. However, the analysis is given as a theoretical discourse to explain the architecture of the protocol.

1. Probability manipulation. Alice is attempting to create a transaction with another risk trying to extend the space of provided addresses for the choice, so she increases her probability to choose a non-payable ticket (for example, Bob proposed 1/100 probability, but Alice is sending back 1/10000 ticket):

   – Will result in the wrong unpacking of the address space “addrchap.” Bob will detect it within the current protocol (See Appendix B “API”).

   In the example above, where Bob took number 115, divided by 10 (115/10=11.5), cut decimals (11.5 ->11) and provided it, Alice, as the address space of the lottery ticket, in normal behavior Alice multiplies 11 by 10 and gets =110, which is the base for guessing.

   To play this lottery, she adds an arbitrary number (n) of the array 0-9. If she randomly chooses 5, Bob wins, because $11\times10+5=115$.

   If she is trying to manipulate with address space, replacing constant 10 with any arbitrary number, say 13, the address space shifts to the range where Bob’s winning is impossible, for instance, $11\times13+n=143+n$. Bob will detect it when Alice sends a drawn lottery ticket because the address will not match the defined range of choices (110,119):

   $143+n \neq (110, 119)$

   Alice may randomly choose a constant (say, 5 instead of 10) and (say, 56 instead of the range between 0 and 9), which leads to the creation of the address in the correct range (110,119), but this manipulation makes no sense. Random “n” either does not lead to the choice of the formally correct address which Bob will detect or leads to the correct address:

   $11 \times 5 + 56 = 111$

   //111 formally satisfies the condition of the range (110, 119): therefore the ticket is playable, even though 56 $\neq (0, 9)$.

   Remarkably, there is a low probability that Alice will randomly pick the address where coincidentally, Bob has a private key, so Bob will get the money, which is not bad at all.

1. Smaller amount. Alice is attempting to send a smaller amount than required. Bob will detect this attack through the analysis of the amount when he receives a draft transaction within the current protocol (See Appendix B “API”).

2. Alice is spending coins that are already spent or non-existing coins. This attack will be checked the same as described in the previous case, in Bob’s wallet when he receives the drawn lottery ticket.

3. An attempt to reuse a signature from earlier used vin[0] in other lotteries. The attack is useless, as it is also detected when Bob verifies “randpay_submittx,” as described in the two cases earlier. Moreover, if Alice creates a naive transaction (does not use RandpayUTXO), Bob can publish it; therefore, permanently burning Alice’s own money.

4.2. Attacks on the network (DS)

These attacks are possible but have adequate preventive countermeasures.
1. Alice is creating two raw DS. The first is a Randpay payment to Bob and the second to herself (or someone else). She is sending both to the mempool.

As a result, there is a logical race. If the Randpay transaction is winning for Bob, he can sign and send it. But if someone has already found a new block with the second transaction which Alice has sent to mempool; the network will not accept Bob’s transaction, because of the nature of the blockchain protocol, which does not allow DS.

Remarkably, Bob will see himself the second Alice’s transaction is accepted in the ledger, therefore, detecting her malicious behavior. So, he will not deliver his services.

Even if Randpay is not winning for Bob, he will see Alice’s second spending in the ledger. Moreover, he can also detect it when it is pending in the mempool.

The earlier Alice creates DS, the higher the chances of a successful attack, but the higher the probability that Bob detects it during the verification.

For the record, there is no way for users to call back their transactions from the mempool. Therefore, Alice will not hide her malicious behavior.

Bob’s tactic here to reduce the risk is to create a random period of delay. For example, it can be around 1-10 seconds to check the existence of DS transaction. In case it appears in the cue of submitted transactions (in the mempool), Bob will avoid sending the product irrespectively.

It should be noticed here that the delay period must be specified based on empirical data of the specific system and business logic. The few-second delay in delivery can be unacceptable for some business schemes, which makes Randpay protocol unusable. To add, if a delay period is applied, this information should not be disclosed. Otherwise, the attacker will try to double-spend after the delay. Also, a good practice here will be if the delay is randomly changed for each transaction within a certain range, it will not be easy for the attacker to detect it empirically.

This vulnerability cannot be excluded at all. Still, Bob can manage his risks by adjusting the pending period to detect the fraud attempt and also can decrease the payment amount. Therefore, the loss will not be dramatic for his business since the attack itself can happen with a certain probability; consequently, it cannot be systematic.

2. Alice is creating DS but will send it only at the moment when she would have seen Bob’s winning transaction in the mempool. In other words, Alice attacks when she sees that Bob wins her money, and so she tries to get it back.

In this case, Alice does not have many methods to get it back; she is seeing a copy of her transaction already signed and sent in the mempool across the blockchain network. At any moment, a new block will be created.

Keep in mind that Emercoin does provide priorities for transactions depending on their fees, all fees (unclaimed output) are burned, miners and minters get their reward only for block creation; therefore, it is impossible to use a higher fee to increase the priority of a specific DS transaction. Hence, this type of attack is impossible.

However, Alice can try to create a circle of malicious nodes around Bob, so when they receive Bob’s transaction, the other segment of the network will not get it. So, Alice will send her DS to the general network. The problem of a node adversary trap is a type of “Sybil attack,” and it is a general problem for any blockchain. The Sybil attack is preventable unless the identity of agents can be verified, and users are prohibited from creating multiple identities or agents (Douceur, 2002).

To decrease the probability of such an attack, Bob can use the command ‘connect’ to communicate only with reliable nodes to avoid direct sending of Randpay transactions to a fraudulent person. Later, when Alice receives Randpay through the blockchain, the lower the chance to win the race. Here, Bob uses the same strategy as described above. When he wins, he waits a while before the provision of the product to Alice.

Furthermore, the level of Bob’s risk can be limited for new customers to avoid setting a high amount on the prize.

Therefore, this attack is possible with a certain probability, and the advice here to Bob is to manage the risks using similar methods in the previous scenario.

3. The client Alice is a PoS minter. Alice may find a kernel transaction that solves the block with DS. However, the block with DS is not sent to the network but used to make a Randpay purchase.

After receiving the product, she publishes the block to the network, thus taking the money back because DS will be confirmed once, while Randpay will not be confirmed at all.

While Alice holds the block privately, there is a chance that someone else will solve it, and Alice will lose her minter’s reward for the block. So, there are two additional options, which are relatively similar to the situation described above:

a) Alice always publishes the held block, thus informing Bob that she is a cheater even if afterward it appears that she would win her money back (with a higher probability); or
b) Alice publishes the block only if Bob’s winning transaction is already on the network, thus removing it from the mempool.

In both cases, the minter Alice has the risk of not getting the reward as a result of the delay.

The average risk is around 1% of the total reward per 6 seconds of delay because the average lag between blocks is 10 minutes. And this affects each block, and not just the winning one.

As a result, the number of losses must be multiplied by the risk rate:

\[
\text{loss} = \text{risk} \times \text{block reward} \times \text{time delay/600s} \tag{5}
\]

Let us take a real-world example of risk analysis. The block award in the block #364069 was 18.388 EMC (Emercoin Blockchain 2019). For instance, the lottery ticket amount (“risk”) is the same 18.388 EMC; if Alice waits at least 60 seconds, her theoretical loss is 18.388∗18.388∗60/600 = 33.8118544 (EMC). It is important to notice that 600 seconds is only an average figure. In this example, the block was found only 3 minutes 41 seconds after the previous. Therefore, the accurate assumption for an attack success is not feasible. Hence, neither Alice can plan her attack more accurately.

To add, Alice needs a lot of coins that have been held at least one month because minting starts after being inactive one month, and the chances increase until the end of the third month and then remain the same; furthermore, such an attack is not always possible because of the random nature of Proof-of-stake consensus (King and Nadal, 2012). In other words, it is not a real option for systematic cybercrimes.

4.3. Recommendations for Bob

The previous subsection shows that:

1) these risks cannot be fundamentally addressed within the current structure of the blockchain protocol;

2) these risks can be addressed using existing protocol by publishing a pre-payment (opening) transaction where Alice’s coins are blocked, but this is plus one transaction that bloats the ledger, and similar schemes already exist, but the condition of Randpay protocol was the absence of any payment channels;

3) these risks have a probabilistic nature and cannot be used for systematic attacks.
Therefore, these risks are considered acceptable regarding the proposed risk management strategy:

1. As in the case with the regular transactions, Bob needs to wait for a couple of new blocks, at least one. By doing so, Bob protects himself from various problems.

2. For real-time sales, when Bob does not wait for published blocks, he should:
   - create a list of existing clients (or at least a list of known IP addresses), not allowing new clients to use higher probability rates;
   - use a random delay in the delivery of the product for new clients;
   - track DS using a different wallet that is not related to his main one;
   - remove unnecessary connections from the main wallet and keep connections only with reliable nodes;
   - if the product per unit is not high-priced, Bob needs to be ready to lose a part of it, like shops lose a part of their assortment as a result of pilfering.

5. Economic analysis and mathematical model

The economic model of Randpay as any other protocol for probabilistic payments is different from the traditional one. It works better for products of a small value which are supplied in a large amount that can be paid by micropayments.

A tariff for phone calls per second is a relevant example here. When each phone call ends with a settlement, it is easy to calculate that with a transaction size of 200 bytes (usually more), 2 trillion world telecom calls per year (Wansink, 2018) will require an annual increase in blockchain size by 400 Terabytes, which is unacceptable for the telecommunication industry.

One can argue that there are mechanisms for reducing the size of transactions, for instance, Segwit, which reduces the size almost by half. But even 200 Terabytes per year instead of 400 is still an insurmountable barrier for the practical use of cryptocurrency in telecom (if the user pays for each call) and in other areas of application that require mass payments.

If Randpay transactions are applied in telecom with the probability 1/10,000 (only one transaction per each 10,000 is paid) the settlement of the world telecom for 2 trillion calls a year will add only 40 Gigabytes to the blockchain, which, although a lot, is acceptable for practical use and can be additionally shortcut by Segwit and the transaction optimizer (Timp, 2019) that will make this figure even smaller.

Using Randpay, clients do not pay small payments for low-value products. There is a small chance that they will lose this lottery and will have to pay the larger amount for the product, but when they regularly consume the product, they on average will pay the same as if they had to pay in the traditional scheme, which is further proved based on the binomial distribution.

The merchant, on the other side, by providing a large number of small value products from time to time, receives large payments at a certain level of probability (which is earlier called “risk”).

These occasional winnings cover a large number of previous losses. Of course, the merchant needs to manage its risks to conduct business in this way carefully.

The use and implementation of this protocol may be confusing. The user may question: “Why must I pay by “lottery ticket” more than I buy (even if a lump sum payment is still small)? Why must I pay the ticket at all when it turns out to be not for me? Why do I not just drop it and refuse to take the product and try my luck to get it for free another time?”

Some consumers may find this payment method inconvenient or unacceptable; therefore, they should find another deal on the market. But when the client Alice agrees and enters into the deal, she must pay even if Bob wins the ticket, because these are the terms and conditions which she agrees on. This is similar to low-cost flights. The tickets are cheap, but if you decide to cancel your trip, you cannot get a refund. That was the deal.

For such payment instruments to work, a user-friendly interface is highly required to make payments smooth. The user experience should be reduced to a simple one-click “Pay” button, which generates a sequence of commands and scripts, including a random choice of the payment address on the lottery ticket.

Since the acquaintance with a probabilistic payment model raises a lot of questions here, it is proposed a mathematical model of its use and influence on the blockchain ledger.

Understanding the economic model of Randpay will help to overcome some of the psychological barriers discussed above.

Below is a simplified statistical model for the set of Randpay transactions based on the average input values. This simplification is correct since transactions in Randpay are mutually independent, and their order does not impact the statistical parameters of the model; that is, it allows arbitrary grouping by any parameters.

The mathematical expectation of profit in the Randpay model is based on binomial distribution, the concept which is developed within the probability theory and statistics (Kaas and Buhrman, 1980).

Randpay is subject to binomial distribution where formula (6) is applied:

\[ E = np \] (6)

where

- \( E \) is an expectation of the sum from \( n \) payments;
- \( n \) is the number of payments;
- \( p \) is the probability.

For the proposed model, this formula can be specified as (7):

\[ E = np \cdot \text{txamo} \] (7)

where \( \text{txamo} \) is the actual amount of payments received by the seller.

The risk here is the initial parameter that the buyer and seller mutually agree on when concluding the contract (8):

\[ \text{risk} = \frac{1}{p} \] (8)

At the same time, the probability is calculated using formula (9):

\[ p = \frac{1}{\text{risk}} \] (9)

The probability of non-winning is calculated respectively using formula (10):

\[ q = 1 - p \] (10)

Therefore, the binomial distribution for a large number of transactions is approximated by the Gaussian normal distribution (Casella and Berger, 2001) (11):

\[ \text{Bin}(n, p) \approx N(np, npq) \] (11)

where

- \( np \) is a mathematical expectation;
- \( npq \) is a variance.

From variance, it is taken a standard deviation (12):

\[ \sigma^2 = npq \] (12)

Therefore, now the deviation can be calculated from the expected amounts. For example, the standard expected amount from \( n \) payments is 1000, but after \( n \) payments with this current risk,
the seller receives the actual amount, which is less, more, or equal to the expectation, shown in formula (13):

$$\sigma = \sqrt{npq}$$  \hspace{1cm} (13)

There is a calculation of a relative error using formula (14):

$$E_{\text{error}} = \frac{\sigma}{E} = \frac{\sqrt{npq}}{np} = \sqrt{\frac{npq}{np^2}} = \sqrt{\frac{q}{np}}$$  \hspace{1cm} (14)

With a lot of Randpay actions, the relative error is solved with the formula (15):

$$\lim_{n \to \infty} \sqrt{\frac{q}{np}} = 0$$  \hspace{1cm} (15)

This equation shows that the relative error for the expected profits tends to be zero. This means in large numbers of transactions; the seller can get what they expect from the business.

The seller's transaction amount is calculated by the formula (16):

$$tx_{amo} = pay_{amo} \cdot \text{risk}$$  \hspace{1cm} (16)

The average fee for transactions tends to decrease when the risk factor increases. This is seen in formula (17):

$$\overline{\text{avg fee}} = \frac{\text{fee}}{\text{risk}}$$  \hspace{1cm} (17)

As it was emphasized that Randpay has a beneficial effect on the standard growth of the blockchain due to normal micropayments, there is a formula (18):

$$\text{blockchain inflation} = \frac{tx_{size}}{\text{risk}}$$  \hspace{1cm} (18)

Finally, let us make some conclusions about the scalability of Randpay within the network.

The initial equation shows that the number of actual transactions that can be inserted in the block is equal to or less than the limit of the block (19):

$$n \leq L$$  \hspace{1cm} (19)

where

$$L$$ is the limit of the block size.

When using Randpay, the transaction amount is not equal to all raw transactions (lottery tickets or deals). They become transactions with a certain amount of probability, which in this case is a 'risk' of the seller (20):

$$n = \frac{n'}{\text{risk}}$$  \hspace{1cm} (20)

where

$$n'$$ is the number of generated raw transactions sent from the buyer to the seller.

Thus, the analysis of the amount of all raw transactions with the current level of the risk looks like (21):

$$\frac{n'}{\text{risk}} \leq L$$  \hspace{1cm} (21)

Eventually, it can be analyzed the number of raw transactions and their influence on the network (22):

$$n' \leq L \cdot \text{risk}$$  \hspace{1cm} (22)

The conclusion here is that for any number of deals, i.e., the total amount of raw transactions, blockchain bandwidth limits can be fitted only with the increase of the 'risk' parameter.

**6. The Lightning Network comparison**

There is no obvious answer as to which micropayments are better. Randpay and the Lightning Network (Lightning Network 2017) have conceptually different approaches with pros and cons.

In the Lightning Network (LN), the payment aggregation system is a separate network of agents connected by payment off-chain channels. In the LN “channel,” Alice or Bob, or both if they have mutual payments, can make an initial blockchain transaction to lock some cryptocurrency as a deposit for their further off-chain interaction.

In the LN, two agents create a channel by linking their cryptocurrency in a ‘channel opening transaction.” They offset each other by not sending the transactions to the blockchain and eventually close the channel by sending a ‘channel closing transaction’ to the ledger. Thus, only two transactions - opening and closing - are reaching the blockchain. Settlement occurs directly between the agents and is not affected by the blockchain network.

LN accommodates the interaction of more than two parties. If Alice and Bob have their payment channel, as well as Bob and Charley, Alice and Charley can make payments through Bob. Bob becomes their payment provider.

Such a system has the following advantages and disadvantages.

**Advantages:**

- High transaction speed because there is no need to wait for confirmation from the blockchain.
- Absolute accuracy of payment because the recipient in each payment receives the exact amount.

**Disadvantages:**

- Requires the creation of processing infrastructure, i.e., a network of operators of channels that are always online.
- The channel operators may want to receive payment for their money transfer services within their channels.
- Payments are possible only in channels; it is impossible to send a payment to any cryptocurrency address.
- Payments become impossible if the channel is not responding, i.e., due to faults.
- When creating a channel, it is necessary to lock some coins in it, and in the case of non-cooperative behavior of either counterparty or denial of service, this amount may be blocked for a long time.
- Denial of service is possible due to the exhaustion of coins in the channel. For example, during mass sales/purchases.
- Protocol with status. Users need to create the channel and store its last state until closing. If it is lost by both counterparties, the money blocked in the channel may be lost forever. Some of these risks are explained by the LN authors (Poon and Dryja, 2016), and other discussions can be found online among enthusiasts (Kelso, 2019; Lemke, 2019; Shaikh, 2019).

Thus, the Lightning Network is similar in purpose and structure to a payment processing system such as VISA or MasterCard in the sense it links holders of crypto wallets. Consider the advantages and disadvantages of the Randpay system compared to the Lightning Network:

**Advantages:**

- A network of channel operators is not required. To work with Randpay, the user needs only a wallet.
- Hence, there are no fees to such network operators.
- Therefore, there are no elements of unreliability associated with the functioning of such operators.
- ‘Any to any’ transactions are possible, what is relevant to the generic idea of a decentralized cryptocurrency.
Table 1
The Lightning Network and Randpay comparison.

| Item | Lightning Network | Randpay |
|------|-------------------|---------|
| 1. High bandwidth | yes | Yes |
| 2. Reduce ledger bloat | yes | Yes |
| 3. Reduce fees compared to regular blockchain transactions | yes | Yes |
| 4. Settlement accuracy | yes | No |
| 5. High speed | yes | Yes |
| 6. Number of blockchain transactions | 2 | 1 |
| 7. Peer-to-peer anyone to anyone (no third parties) | no | Yes |
| 8. Need infrastructure | yes | No |
| 9. Restrain funds | yes | No |
| 10. Requires channel | yes | No |
| 11. Fees to a channel operator | yes -(no channel) | -(no channel) |
| 12. Risk of running out of money in the channel | yes -(no channel) | -(no channel) |
| 13. Risk of loss of channel control | yes -(no channel) | -(no channel) |
| 14. Need to keep the state (of the channel, offset) | yes -(no channel) | -(no channel) |
| 15. Rates can be smaller than the smallest payment unit (less than 1 Satoshi) | no | yes |
| 16. A short period of the settlement | no | yes |
| 17. Implementation | Developed, the early stage of use and testing | Developed, testing |

- The actual rate of goods can be even below the minimum cryptocurrency unit (Satoshi). For example, 1 second of a call can cost 0.5 Satoshi.
- Funds are not restrained in channels for a long period. It usually becomes clear in seconds whether the ticket won or not, and the payer can reuse non-paid coins in any other payment.
- There is no denial of service problems due to money exhaustion in the channel.
- Compared with two opening and closing transactions in the Lightning Network, Randpay sends one transaction to the blockchain, which makes the system twice as efficient with the same aggregation factor.
- Randpay is a stateless protocol. No need to establish a channel or other financial relations with the counterparty, keep the state of the channel, and close it correctly. Therefore, there is no risk of the channel being lost.
- The special completion of the payment protocol is also not required, which is especially useful with an unreliable Internet connection. Either counterparty can disconnect at any time without harm to payments.

Disadvantages:

- For a reliable payment, one needs to wait for the confirmation of the transaction by closing the blocks, as in regular cryptocurrency payments. Work without confirmation is possible but requires additional measures to protect against fraud.
- The actual amount of payment will differ from the ‘fair’ price, but in the long run, it will tend to it as it is shown in formula (15).

A brief comparison of Table 1 can be presented as follows:

7. Conclusions

The research and tests show that Randpay is a sustainable technology for micropayments.

The model is based on the buyer’s desire to gain the product for free and acceptance of the possibility of having to pay a higher price from time to time, but which tends to become a fair price during regular use for both parties.

The seller is a supplier of low-priced products. In traditional business, it typically leads to high transactional costs, either in cash payments or cryptocurrency. The model offers the seller a certain level of probability to receive a lump sum from time to time instead of micropayments.

The seller decides for themselves the level of probability (risk). Of course, all of this makes sense for large numbers of transactions, which increases the chances for lump sums. Over an extended period, the proper risk/probability averages profit to the level of the traditional model of business. One-time customers are also possible because the profit is calculated through all transactions of all customers of the product.

Also, the seller can choose a strategy of selling simultaneously using both models: Randpay and traditional subscription model, which gives a competitive advantage in the market before those buyers that, for some reason, may want to use Randpay. For example, some people are gamblers, or because of a typical consumer’s desire to get free stuff.

The implementation of this protocol required the research work even though it is based on some known concepts like Rivest’s ‘lottery ticket’ and Blum’s ‘flipping coins by telephone.’ In the findings of the research, two original solutions were found: ‘addrchap,’ which is based on Blum’s tool for creating a provable and fair peer-to-peer remote off-chain lottery ticket where the choice is offered as a space of payment addresses with only one payable, and the second finding - RandpayUTXO - which is an unspendable zero output as an addition to the blockchain protocol that requires the payer’s signature on the payer’s transactions to prevent locking the funds on a dead-end address.

In the results, it is also concluded that RandpayUTXO has a side effect beyond the microtransactions protocol because it can be used for regular transactions for which the law requires the explicit consent of the recipient to be recognized as valid.

Randpay has a beneficial effect on the network. While the capacity of the blockchain protocol remains the same, the network, in general, can provide for better performance in terms of the number of transactions that users can conduct in their business because only a part of these will go to the network. Randpay also helps to speed up payments. Randpay here helps to overcome the inherent average time of blockchain transaction acceptance by offering parties to conduct most of their transactions on the blockchain instantly (or at very high speed), which is comparable to the average of 10 minutes. The use of such a tool reduces the average fee per transaction, which, along with other conveniences, in general, reduces transactional costs in a broad sense.

Author contributions

Oleg Khovayko is an author and developer of the Randpay protocol. Oleksii Konashevych is a researcher who provided the analysis of the protocol and outlined the results in the paper. Co-authors contributed equally to writing this paper.

Declaration of Competing Interest

None.
Appendix A

Randpay testing

1. Script:

```bash
#!/bin/sh -v

echo 'Randpay test started'

# 1st step: create a challenge with risk=3 (probability=1/3) and 1000s timeout
CHAP=`emc randpay_createaddrchap 3 1000`

# 2nd step: create RandpayTX for 0.5EMC, chapaddr=got_above, risk=3, 1000s timeout, non-naive
TX=`emc randpay_createtx 0.5 $CHAP 3 10000`

# 3rd step: submit to the wallet the RandpayTX(got in 2), with risk=3. Wallet returns lottery submission result and signs if the ticket is winning.
emc randpay_submittx $TX 3
```
II. Tests
For this research, the script ran seven times. Two winning tickets were obtained. The raw transaction unpacked and verified successfully, signed, and published in the testnet. The mathematical expectation of the payment for ‘0.5 EMC ticket’ per transaction was: $0.5/3 \approx 0.167$ (EMC). The actual result was $1/7 \approx 0.142$ (EMC).

III. Resulting transactions
Testing results are recorded in Emercoin testnet in the following Randpay TXs:

1) TX = 7a4e7e92b6287e77c02d47265f1a2e8c0206b4efef67a010d3050c3dfcfbc16f3
2) TX = 52a2d14e9f70d6d2c3f8aea81b6748ec688f3c5c0c19c993fee9fa3c3223c23

Details are also available in online explorer:
http://testnet.emercoin.com/tx/7a4e7e92b6287e77c02d47265f1a2e8c0206b4efef67a010d3050c3dfcfbc16f3
http://testnet.emercoin.com/tx/52a2d14e9f70d6d2c3f8aea81b6748ec688f3c5c0c19c993fee9fa3c3223c23
Winning transaction are published in the test ledger. The following sections presents resulting RandpayUTXO transactions in JSON:

1) getrawtransaction 7a4e7e92b6287e77c02d47265f1a2e8c0206b4efef67a010d3050c3dfcfbc16f3 1

(continued on next page)
2) getrawtransaction 52a2d14e9f70d6d2c3f8ea8a1b6748ec688f3cc5c0c19c993fee9fac3223c23 1

```json

"addresses": [  
  "e0RxVvW1LSRoIAeQeGDMTeOXSggfU2LpCwr"
  
},

"value": 89.2979900,
"n": 1,
"scriptPubKey": {  
  "asm": "OP_DUP OP_HASH160 4d3e7906b623a2d7a76b366c999e523e5f72 OP_EQUALVERIFY OP_CHECKSIG",
  "hex": "76a9144d03e7906b623a2d7a76b366c999e523e5f7288ac",
  "sigHash": 1,
  "type": "pubkeyhash",
  "addresses": [  
    "mn/YB17uE8j38rgwX64S3jyrAFDXeUzBy"
  
}

"blockhash": "133c1cbe4009026602c9742907138aaa05e89e9e631f6e04e26d155b8814",
"confirmation": 67970,
"blocktime": 1546881734

```

(continued on next page)
Appendix B

Randpay API

Randpay API is implemented as an extension of the JSON HTTP Emercoin API and provides for the generation and processing of Randpay transactions by a node/wallet. Using this API, the user’s program (for example, Bob’s selling platform) interacts with the node and makes or receives payments. In the case of the correct operation of the API functions, the node returns values indicated in the specification below. In the case of incorrect data input, malicious behavior, or error run, a standard error code is returned with an accompanying text message.

A Steps

Randpay payout protocol (Emercoin kernel from version 0.71 (Emercoin Github 2018)) consists of the following steps¹:

1. WANTPAY(double amount, uint32_t risk).

2. WANTPAY(double amount, uint32_t risk).

---

¹ Please consider this as a pseudo programming code for a short illustration in this paper. Consult the Emercoin core code on the page of the project: https://github.com/emercoin/emercoin.
This step is optional yet desirable for bargaining. By sending this request, Alice informs Bob that she is interested in a Randpay payment of a certain amount with the probability to pay \( p = \frac{1}{\text{risk}} \). In other words, the real average amount of the onetime payment is amount/risk.

For example, in the case of WANTPAY(100, 100000), Alice informs that she wants to pay 100 EMC with a risk rate of 1/100000, i.e., the real averaged amount of the payment is 0.001 EMC.

The example can be the following: Alice buys the package 1 second of phone calls for 0 EMC or 100 EMC, where 100 EMC is what she pays with the probability of \( 1/100,000 = 0.00001 \). With the regular purchases according to the binomial distribution, she pays 100/100,000 = 0.001 EMC per second (see Section 5).

3. NEEDPAY (double amount, uint32_t risk, char [] addrchap)

Normally, Bob's NEEDPAY request is sent as an answer to Alice's WANTPAY in the previous step; however, NEEDPAY is obligatory.

The businessman Bob informs client Alice that he would like to receive payment of a certain amount for his services with the probability of \( p = \frac{1}{\text{risk}} \), thus offering the client the chance to guess an address from the space of the addresses (risk, addrchap).

Parameters (amount, risk) sent to the client can be different from those indicated in WANTPAY request. This is made to allow Bob to refuse risky offers.

4. PAYMENT (uint32_t risk, char[] rawtx)

Client Alice sends a Randpay transaction to Bob's server following the parameters set in NEEDPAY.

First, Bob checks the amount on his own, if the chosen amount by Alice's payment/risk satisfies him, then she sends it to the wallet for verification using the blockchain protocol. In case there's no sign of malicious behavior (see the next subsection for details of attack scenarios), Bob can provide the services to Alice.

If Bob wins, the wallet will sign the transaction automatically and publish it in the blockchain of Emercoin.

A API elements

API consists of the following elements:

- randpay_createaddrchap (uint32_t risk, int timeout)

  Bob creates a pair of private and public keys for the given probability rate and generates the base (the raw) of the space of addresses 'addrchap' from the public key:

  \[
  addrchap = \frac{\text{hash}160(\text{Pubkey})}{\text{risk}}
  \]

  The 'addrchap' result (160 bits in HEX) is used for NEEDPAY step (see below) and makes no changes in the blockchain. However, the wallet keeps the generated values in memory for further use in a pair 'addrchap -> Privkey'.

  'Int timeout' is the parameter of the period during which the user wants to cache this data.

- randpay_createhex (double amount, char[] addrchap, uint32_t risk, int timeout, bool naive=false)

  Payer Alice creates the raw (draft) transaction with the 'amount' and a selected address during which she must:

  a) unpack 'addrchap' from hex into the binary form

  b) create an address 'rand_addr' from the pair (risk, addrchap)

  \[
  \text{uint160 rand_addr} = \text{risk} + \text{addrchap} + \text{GetRand(risk)}
  \]

  c) create a Randpay transaction to rand_addr; it is obligatory to put the payment into vout[0] and use vin[0] to indicate RandpayUTXO

  d) put the used inputs 'on hold' not to accidentally spend them in other payments while Randpay is not finalized (around 30-60 seconds)

  e) signs all the available inputs (except RandpayUTXO)

  f) choose parameter 'naive' if the payment does not require RandpayUTXO, as otherwise randpay-in must be added and the spending declared to the vin[0]

  System returns: randpay_tx_hex // HEX raw transaction;

  Alice sends 'randpay_tx_hex' to Bob.

  - randpay_submittx(char[] Randpay_tx_hex, uint32_t risk)

  Bob's wallet verifies here the code of the draft transaction using the previously cached pair 'addrchap -> Privkey.' When Bob wins, he signs a transaction and sends it to the network.

  Bob's wallet steps:

  a) Check signatures for inputs for the given raw_tx_hex, except for randpay-in 'vin[0]'. If a signature is missing or if the input 'vin[0]' is signed, then check it also. If at least one of the inputs is not available, not signed, or incorrectly signed, it returns error 'err=-1' (double-spending attempt). Correct and winning transactions are passed on to step (b). Otherwise, they progress to step (c).

  b) Sign RandpayUTXO vin[0] with the private key, associated with vout[0], and send the transaction to the blockchain (winning case).

  c) If Alice does not guess the address, then Bob is unable to sign RandpayUTXO, and this transaction (non-winning case) is just dropped.

  System returns:

  - amount – the sum from vout[0];

  - won - Boolean indication of a winning ticket.
Appendix C

**Randpay URI**

Emercoin wallet contains a mechanism for sending Randpay payments by calling it via a URI by an external application (for example, a browser), similar to the Bitcoin BIP21 mechanism ([Bitcoin URIs, 2019](#)), which Emercoin also supports. This mechanism allows internet sites, through the user’s browser, to request micropayments in Emercoin, forming a corresponding URI on the page. The user clicks the link, and the user’s wallet after confirmation immediately pays the site through the Randpay mechanism.

This interface can be used by sites to sell access to articles or media content (video, music, etc.).

For example, a site may request 0.001 EMC for accessing an article by creating a Randpay request with the parameters `amount=10` and `risk=10000`. An example of a URI for such a request is the following: `emercoin:randpay?amount=10.0&chap=00deadbeef&risk=10,000&submit=http%3A%2F%2Frandpay.news.com%3Fid=777%26article=666`

Let us explore more the Randpay URI structure. Its general appearance is: `emercoin://randpay?amount=DOUBLE&chap=chap_hex&risk=INT[&timeout=INT]&submit=CALLBACK_URI`

Here in square brackets are elements of the URI, which the site may not specify. The URI parameters of this interface (amount, chap, risk, timeout) correspond to those in the `randpaycreatetx()` call.

- **amount** – the amount of the payment transaction which Alice will pay if Bob wins the lottery.
- **chap** – the requirement of a lottery ticket in the hex representation, formed by the recipient (website).
- **risk** – the reciprocal of the probability of winning.
- **timeout** – the time interval in seconds for which the UTXO outputs that are involved in the transaction are blocked.

The parameter `submit` contains a `callback-URI`. There, the user’s wallet will send the generated Randpay transaction (lottery ticket) in HEX code using HTTP POST. The special characters contained in this parameter reserved in rfc3986 standard ([Masinter et al., 2020](#)), for example, `/` & `*` must be recoded into a ‘percentage representation,’ to avoid a conflict between the parameters of the original URI and the callback-URI.

The user of the wallet can specify the parameters controlling the behavior of this interface in the `emercoin.conf` file (after the `=` sign, the default parameters are shown):

- `rp_max_amount=0` – the maximum amount in a transaction that can be sent without confirmation, EMC.
- `rp_max_payment=0` – the maximum amount of payment that can be sent without confirmation, EMC.
- `rp_timeout=30` – blocking time of inputs used in a Randpay transaction, sec.
- `rp_submit=false` – automatic selection of the ‘Submit’ button in the payment confirmation window. The default choice is CANCEL, that is, cancellation of payment.
CRediT authorship contribution statement

Oleksii Konashevych: Writing - original draft, Methodology, Formal analysis, Visualization. Oleg Khovayko: Conceptualization, Software, Methodology, Validation, Supervision.

References

Ali, S.T., Clarke, D., McCorry, P., 2017. "The Nuts and Bolts of Micropayments: A Survey." Arkiv, 09-Oct-2017.

"Ardor." [Online]. Available: https://ardordocs.jelurida.com/Getting_started. [Accessed: 07-Feb-2019].

"Bitcoin address • Programming The Blockchain in C#." [Online]. Available: https://bitcore.io/api/lib/uri. [Accessed: 15-Jan-2019].

Blum, M., 1983. Coin flipping by telephone a protocol for solving impossible problems. ACM SIGACT News 15 (1), 23–27 Jan.

Casella, G., Berger, R., 2001. Statistical Inference. Popul. Trends.

Chiesa, A., Green, M., Liu, J., Miao, P., Miers, L., Mishra, P., 2017. Decentralized anonymous micropayments. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT 2017), pp. 609–642.

Dai, X., Chaudhary, K., Grundy, J., 2007. Comparing and Contrasting Micro-pay Models for Content Sharing in P2P Networks. In: 2007 Third International IEEE Conference on Signal-Image Technologies and Internet-Based System, pp. 347–354.

Douceur, J.R., 2002. The Sybil attack. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics).

Elshenkih, M., Youssef, A.M., Clark, J., 2019. Deploying PayWord on Ethereum. Financial Cryptography 2019: FC’19.

"Emercoin." [Online]. Available: https://emercoin.com/. [Accessed: 10-Nov-2017].

"Emercoin Blockchain." [Online]. Available: https://explorer.emercoin.com/block/97f633dca4370859720c3e5e3eb001c57f1545a0896432643604b02d4a27e633. [Accessed: 08-Jul-2019].

"Emercoin Github." [Online]. Available: https://github.com/emercr. [Accessed: 05-Sep-2018].

Hu, K., Zhang, Z., 2018. Fast-lottery-based micropayments for decentralized currencies. In: Australasian Conference on Information Security and Privacy (ACISP 2018), pp. 669–686.

Kaas, R., Buhrman, J.M., 1980. Mean, median and mode in binomial distributions. Stat. Neerl. 34 (1), 13–18 Mar.

Kelso, E., “Lightning Network Warning: Please Don’t Lose Money, Learn from My Recklessness - CoinSpice.” [Online]. Available: https://coinspice.io/news/lightning-network-warning-please-dont-lose-money-learn-from-my-recklessness/. [Accessed: 08-Jul-2019].

King, S., Nadal, S., 2012. "PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake". Kou, W., 2003. Micropayments. In: Payment Technologies for E-Commerce. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 245–262.

Lemke, P., "How to backup your lightning network channels – Patrick Lemke – Medium." [Online]. Available: https://medium.com/@patricklemke95/how-to-backup-your-lightning-network-channels-170c995c157b. [Accessed: 08-Jul-2019].

“Lightning Network.” [Online]. Available: https://lightning.network. [Accessed: 26-Jan-2017].

L. Masinter, T. Berners-Lee, and R. T. Fielding. “Uniform Resource Identifier (URI): Generic Syntax.” 2005.

"Micropayments Overview," W3C. [Online]. Available: https://www.w3.org/ECommerce/Micropayments/Overview.html. [Accessed: 24-Jun-2019] 2019.

Nakamoto, S., "Bitcoin: A Peer-to-Peer Electronic Cash System." 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf. [Accessed: 27-Dec-2016].

Ni, P., Li, H., Pan, D., 2018. Micropaying to a distributed payee with instant confirmation. In: International Conference on Information and Communications Security (ICICS 2018), pp. 793–804.

Papaefthathios, I., Manifavas, C., 2004. Evaluation of Micropayment Transaction Costs. J. Electron. Commer. Res. 5 (2).

Pass, R., Shelat, A., 2015. Micropayments for Decentralized Currencies. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security - CCS ’15, pp. 207–218.

Poon, J., Dryja, T., 2016. “The Lightning Network: Scalable Off-Chain Instant Payments” Online. Available: https://lightning.network/lightning-network-paper.pdf. [Accessed: 19-Jan-2017].

Rezaeibagha, F., Mu, Y., 2019. Efficient Micropayment of Cryptocurrency from Blockchains. Comput. J. 62 (4), 507–517 Apr.

River, R.L., 1997. Electronic lottery tickets as micropayments. In: FC 1997: Financial Cryptography, pp. 307–314.

Salamon, D.L., et al., 2018. Orchid: enabling decentralized network formation and probabilistic micro-payments. Orchid.

Shahk, R., “Enable Channel Backups and Fund Recovery on LND - Lightning Network.” [Online]. Available: https://medium.com/@rahil471/enable-backups-and-fund-recovery-on-lnd-lightning-network-3f27be42ebf4. [Accessed: 08-Jul-2019].

“Sharding FAQs On sharding blockchains.” [Online]. Available: https://github.com/ethereum/wiki/wiki/Sharding-FAQs. [Accessed: 13-Jan-2019].

Timp, J., “Emercoin Implements Solution To Reduce Blocksize Inflation.” [Online]. Available: https://cointelegraph.com/news/emercoin-implements-solution-to-reduce-blocksize-inflation. [Accessed: 16-Jan-2019].

“Understanding Segregated Witness – SegWit Resources,” Sew. [Online]. Available: https://segwit.org/understanding-segregated-witness-905cc712cf92. [Accessed: 21-Sep-2018].

Wansink, K., “Global Telecoms - The Big Picture 2019 - Key Industry Statistics,” 2018. Zhong, L., Wu, Q., Xie, J., Guan, Z., Qin, B., 2019. A secure large-scale instant payment system based on blockchain. Comput. Secur. 84, 349–364 Jul.

Oleksii Konashevych Erasmus Mundus Joint International Doctoral Fellow in Law, Science and Technology (European Union). Degrees in Laws (2005) and Economics (2010). Research in blockchain, e-governance, e-democracy and real estate, http://oleksii.konashevych.com.

Oleg Khovayko MS in CS, National Research Nuclear University MEPhI (1994), has a background in the realm of IT, biotech, finance and specializes in the technical aspects of blockchains and cryptography, Tower Architect of Emercoin, https://emercoin.com.