Mohsen Rahmanikivi *et al*, International Journal of Computer Science and Mobile Applications,
Vol.7 Issue. 10, October- 2019, pg. 1-11 **ISSN: 2321-8363**
Impact Factor: 5.515

Introducing Two-Factor Authentication for the Block Chain Protocol with help of Hash Algorithm

Mohsen Rahmanikivi[1]; Parisa Daneshjoo[2]
[1]Islamic Azad University West Tehran Branch,Tehran, Iran, m.rahmani724@gmail.com
[2]Assistant Professor, Department Of Computer Science and Engineering, Islamic Azad University West Tehran Branch,Tehran, Iran, daneshjoo.p@wtiau.ac.ir

Abstract

In the blockchain structure, the creator has used the Elliptic Curve Digital Signature Algorithm (ECDSA), which provides public and private keys for users. The Users use private keys to create digital signatures in order to generate transactions in the blockchain network, and the only authentication method for the transactions is to verify the digital signature using public keys. Another use of public keys also is creating wallet address. This verifying method causes some problems over private keys that the users have to protect their private keys against threats, lest a thief steals all their assets by stealing their private keys. Because of this, there are many ways to protect private keys, most of which emphasize protection by the users, with a variety of wallet solutions to protect private keys such as storing private keys in encrypted files , storage devices, and physical wallets. In this article, we try to introduce a new two-step authentication method to provide more security, accessibility, and convenience. We will discuss about a new method which provides a one-time password (OTP) or stable password to create transactions, so it provides a new blockchain structure with the help of the hash algorithm. In this structure, The first authentication step is the regular one which is the verifying digital signatures by the public keys, and the second step is a decentralized verification of the password. Such an authentication method is not depended on any wallet and executed directly by every node in the network with the help of data saved in the blockchain like verifying digital-signature.

Keywords*:* Blockchain, Bitcoin, Two-factor authentication, one-time password (OTP), stable password.

1. Introduction

Blockchain [1-3] [4] [5] is a distributed, decentralized database which can control the accuracy of the information within itself. It operates peer to peer and does not require any central policymaker or central controller. The first cryptocurrency, bitcoin, and the Blockchain structure were introduced in the year 2008 by Satoshi Nakamoto [6], Bitcoin is based on the Blockchain network, and designed to meet the two essential criteria of electronic money[7] that only the money-holder can spend and cannot spend twice.
To create these features, it uses the blockchain database. This database contains a chain of interconnected blocks, each of which also contains a chain of transactions, and Bitcoin fulfills the two criteria mentioned by checking the chain of transactions and reviewing all history of every transaction and related keys in the database.

Before a transaction has occurred, each user needs to create two keys for him/herself using the Elliptic Curve Digital Signature Algorithm (ECDSA) [8], , a private key, and a public key. The address of each user is obtained from their public key and is used to receive the transaction. Each transaction on the network includes the sender and

the receiver public keys, and the sender creates transaction by signing the data of transaction using his private key [3].

The private key is the only member that is not propagated on the network, and the only effect of that is the digital signature [9] created in the transaction, and any private key holder can use it to create the signature to execute a transaction in order to transfer the bitcoins related to that private key.From a thief's point of view, only by stealing the private key, everyone can acquire bitcoins related to that key. This is the reason why users know that their private key is as important as the value of their bitcoin and why they must protect it.

To protect the private keys by the users, they usually try to hide them in encrypted files or store them in wallet devices which are needed some conventional security method with a lot of security problems[10]. Moreover, there are solutions to allow users to provide more security, similar to those which are available in the banking system[11].

Supposing the private key is like an ATM card [12] , typically, in the bitcoin network, and every transaction, users use their card (=private key) for creating a transaction, and users perform it without need of stating any password whereas in the conventional banking system, it is mandatory to state password. In order to simulate the banking system authentication method, three ideas have been proposed in the literature; namely, Multi-signature [13] and two-party ECDSA signature[14], which are discussed in detail hereinafter.

1.1 Multi-signature support in Bitcoin

Bitcoin's structure allows us to use multi-signature transactions[13] instead of just one signature for the transaction, so that we can record t signatures for the outgoing transaction then these t signatures must be verified by u public keys which have to be allocated in the previous transaction. In this structure, the sender has to use u public keys in the transaction, so if the receiver wants to transfer his bitcoin, he must consider t signatures for the new transaction, each of which will be verified by u mentioned public keys.

Due to the multitude of public keys, a single, short, unique wallet address cannot be created, so to solve the problem of the multitude of public keys, it is proposed to use scripting function and pay to hash feature [15]. Instead of using u public keys, we use a hash and an address derived from the hash which this hash comes from the script that contains a list of u public keys. So the sender can use it instead of a list of public keys, then the receiver still must use t signatures to create a new transaction, and each of signatures must verify with the list of u public keys which is stored in the script.

The advantage of this method is, the transaction can be executed on the current Bitcoin network, and it requires multitude private keys to create the transaction so that we must have all them because the transaction needs the whole keys to be created. Problems with this structure can be divided into two parts. The first part includes the answer of how to protect private keys and keep them in a safe place, with high accessibility, and avoiding to lose or miss keys because losing a key equal to all money lost. The other part is over the complexity of the execution which needs a lot of training before using, but our two-factor authentication is straightforward to use and do not need multitude keys.

1.2 Two-party ECDSA

PhilipMacKenzie1, Michael K. Reiter2 [16] presented Two-party DSA signature, and they showed that instead of using the private key to create the signature, we could create two parts derived from the private key and introduced multiparty DSA signature. They said this sentence to introduce their work "that two parties can efficiently generate a DSA signature with respect to a given public key but neither can alone.[16]"

Christopher Mann1, Daniel Loebenberger1 [14] developed the two-party DSA signature to the two-party ECDSA signature to be compatible with bitcoin structure. Then they used it to create two-factor authentication. Therefore instead of creating signature depended on a private key, they used Two different shares of the private key, which are independent and not capable to signing alone, so these two parts have to work together to create the signature. The collaboration between two applications that store two parts build the signature, for example, an app in a computer and another one.

This method focuses on creating a typical transaction although The only difference with the usual structure is how to create a transaction by the user. So as an advantage if someone looks at the blockchain network, they will not be noticed who uses this signature method. Another advantage of this structure is the need for efficient communication of two-part carriers, And it eliminates the possibility of stealing the private key because there is no unified private key. The disadvantage of this structure is the complexity of using for regular users. Every new user has to obtain two devices and be trained to understand how to use, it is expensive and likely to fail and if a user loses one of the two parts because of lack of knowledge that eventually leads to the missing money.

2. Problem and Solution

We need to protect our private keys In Multi-signature[13] and two-party ECDSA signature[14]. In the first one, we need to protect multi private keys, and in the second one we need to protect two shares of the private key, and the losing of one key or one share is equal to lose all assets that we have in the blockchain network. Also, those methods aggravate the problem which is how to keep our private key in a safe place because, before them, we just need to protect from one private key, but by using these methods we have to concern about keeping multi private keys or two shares in the safe place.

The practical way to increase security and avoid complexity is the use of regular banking structure of the debit card, so if we consider our private key as a debit card, then we should be required a password for every transaction to be the same as the banking system. Also, this password should be verified decentralized because it must not conflict with features of BlockChain; finally, A transaction does not be executed without the correct password. This password is the second step of authentication, and as an additional feature, any file such as a music file, a photo, data or document or any character would be used as a password without any limitation.we will discuss both onetime password and static password as the second step of authentication in section \ref{sec:1} and section \ref{sec:6} respectively.

3. Second step of authentication with One-time Password

In this structure, we use two hash-digest which is stored in the transactions to enable the second step of authentication. We call them, Pass-Digest, and Pass-Verifier which we will discuss in detail in section \ref{sec:3}.However, the user states a password (one-time password OTP) at the time of sending bitcoin which his/her pass-verifier was generated at the time of receiving the bitcoin. It would be a better way to discuss regular transactions, and after that, show how Two-step authentication transactions work.

3.1 A simple view of a regular blockchain transaction

Every transaction has a sender and a receiver, in this sample, user1 is a sender, and user2 is a receiver. user1 intends to create a transaction included a specific context (it might be a mention for an amount of bitcoin) to user2, after publishing that user2 will have been the owner of the context and can manage it or send it to anyone, as user1 did.Therefore, creating the transaction needs to have user1's private key, namely PrivateKey1 and User2's PublicKey called PublicKey2. To keep the structure evident and straightforward, we skip from other data needed to create transaction such as previous transaction digest, practical data, and use the public key as wallet address. Finally we need a digital signature created by privatekey1 As it is shown in Fig.1 in order to sign the file which includes the context.

All Requirements for creating a regular transaction are:

1- PrivateKey1 for creating the digital signature

2- PublicKey2 for being the destination of the transaction (wallet)

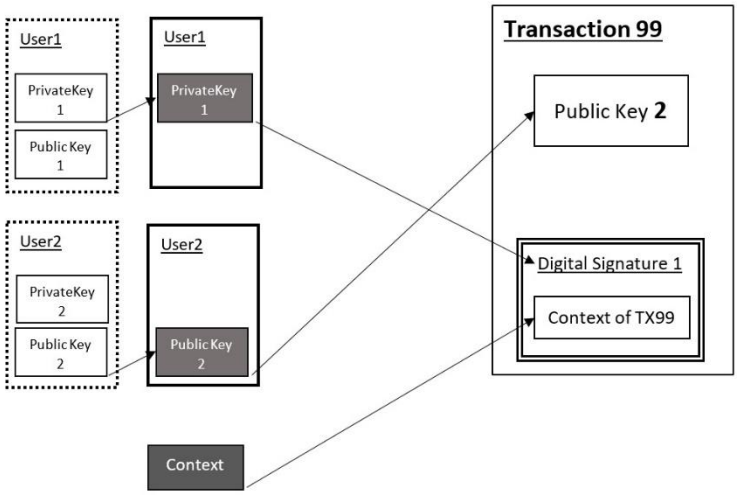

Figure 1: *A simple view of a regular blockchain transaction*

3.2 A simple view of a Two-step authentication transaction

user1 intends to create a transaction with a specific context to user2, as a regular transaction but in an especially secure way. This new kind of transaction is protected with a password which is checked in the decentralized way to verify the current transaction. The transaction needs user1's PrivateKey and user2's PublicKey, but both are not sufficient. The transaction also needs PassDigest of user1 and PassVerifier of user2. To create the transaction we put PassVerifier of user2 and PassDigest of user1 into the transaction (see Fig.2)

3.2.1 What is PasDigest?

PassDigest (1) is a digest coming from the output of Secure Hash Algorithm 256(SHA256) which the algorithm input is the password (we will discuss how to chose this algorithm in section. There are two reasons for creating it, the first, making the plaintext of the password unreadable and the second is making it fixed-size which is unbiased to diversity of password characters.

PassDigest =Hash("Password") (1)

If Password is "12345" PassDigest is :

=Hash("Password")

=SHA256("12345")

="5994471abb01112afcc18159f6cc74b4f511b99806da59b3caf5a9c173cacfc5"

3.2.2 What is PassVerifier?

PassVerifier (2) is a digest which we can verify the password(and PassDigest) with the help of it; it is made by executing the same hash algorithm with using PassDigest and PublicKey as inputs.

PassVerifier= Hash ("PassDigest"+"PublicKey") (2)

If Password is "12345"

PassDigest is :

="5994471abb01112afcc18159f6cc74b4f511b99806da59b3caf5a9c173cacfc5"

PassVerifier is :

=SHA256("5994471abb01112afcc18159f6cc74b4f511b99806da59b3caf5a9c173cacfc5"

+

"04cb3717d143e1addcd50c4810aa7f63124e6f2abdd03eebad01a2501e1c44059945acc88a3254469fa6fb49e905702cf987d2064863ece2e1485813609c807245")

="6568aadfcd0dab3d94ffcaeb9ff5047317edefc24904c8bd356795098de4bb86"

To Create a Two-step authentication transaction, first, user2 mind a password (The password is a OneTimePassword (OTP) because the user has to state a new one for every transaction and it is just secure for the current transaction.) and creates his/her PassDigest (3) and PassVerifier (4).

PassDigest2=Hash(Password Of User 2) (3)

PassVerifier2=Hash(PassDigest2+PublicKey2) (4)

In the second step, user1 who chose his password by stating his/her PassVerifier1 in the previous transaction should state his/her password (=PassDigest) (5) in terms of answering the required password for approving the transaction which is performed by blockchain node.

PassDigest1=Hash(Password Of User 1) (5)

Now we have all requirements for creating a two-step authentication transaction which are put into transaction, as you can see in Fig.2.

1- PrivateKey1 (for creating the digital signature)

2- PassDigest1 (for second step of authenticating)

3- PublicKey2 (the destination of the transaction)

4- PassVerifier2 (for verifying next transaction)

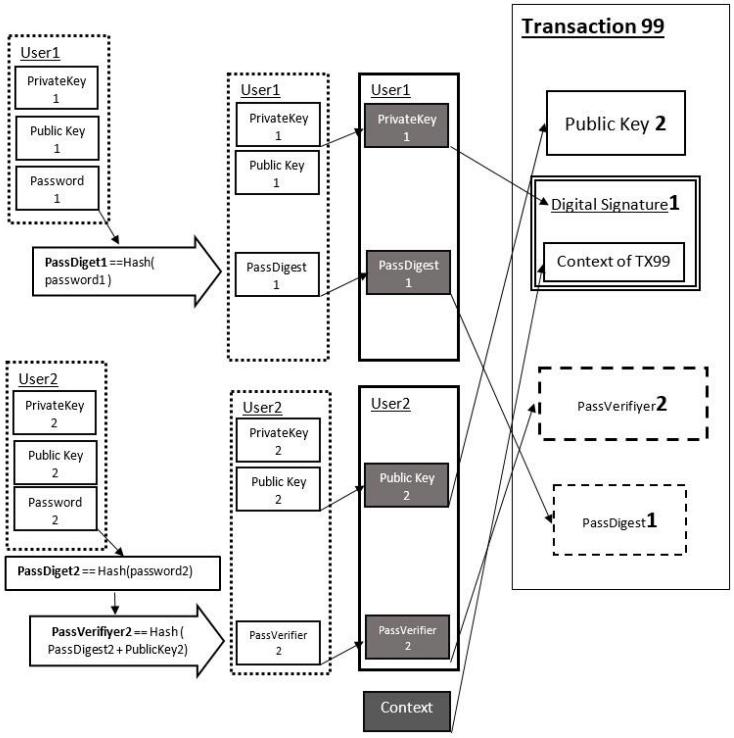

Figure 2: *A simple view of a Two-step authentication transaction*

3.3 How distributed verifier work

To Verify PassDigest, The nod tries to recreate PassVerifier (6), therefore collect information from received data (PassDigest) and fetch information from saved data in the previous transaction (public key). Then create a new PassVerifier from collected data, it does this by adding the public key at the end of received PassDigest, then execute the hash algorithm with this data as the input.

$$PassVerifier=Hash(PassDigest+PublicKey) \quad (6)$$

After creating the new PassVerifier, the node compares (7) it with the original one in the previous transaction, if both are same, the nod approves the transaction.

$$PassVerifier != NewPassVerifier \quad (7)$$

In our example , we see that the user1 uses his/her private key and the public key of thr user2 to create the transaction,then the user1 put his PassDigest and PassVerifier of the user2 into the transaction and send it to network, Finally, every node received this transaction approves it if both digital signature (step one of authentication) and PassDigest (step two of authentication) verified correctly.what we can see in Fig.3

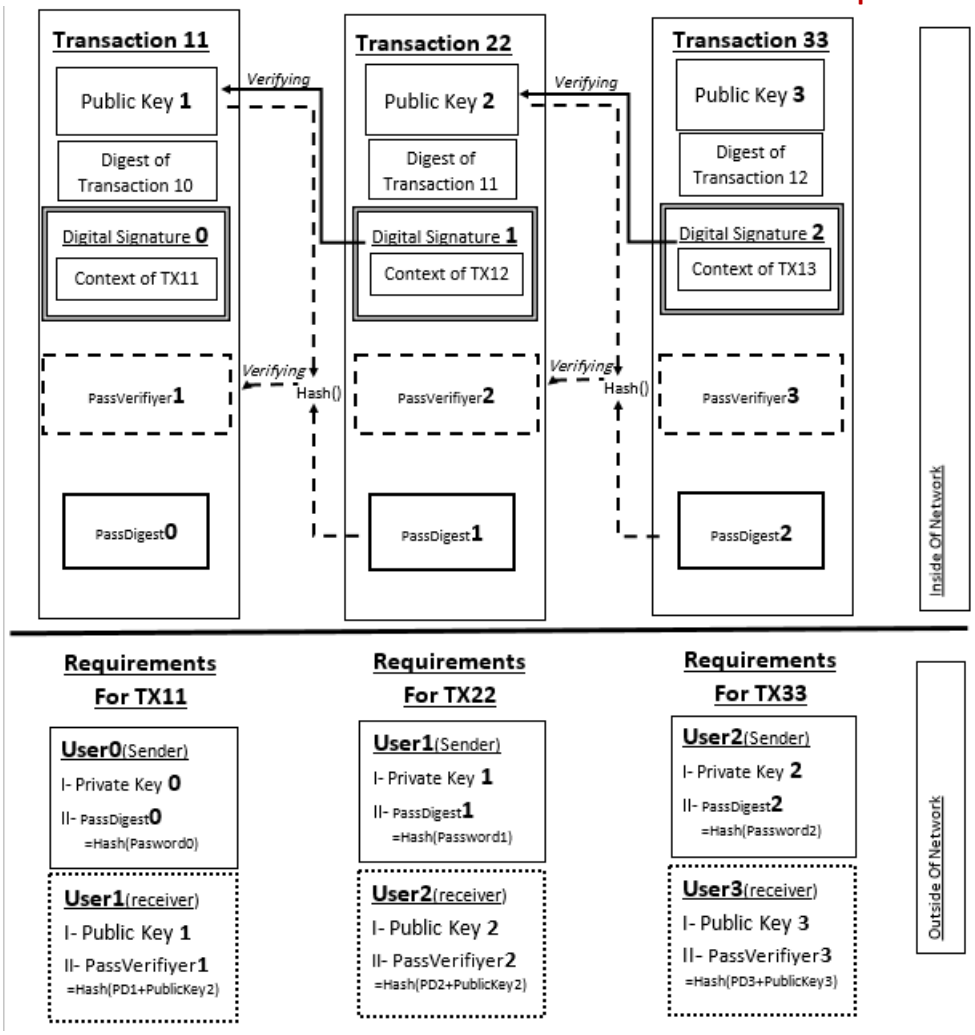

Figure 3: *How distributed verifier work*

3.4 Second step of authentication with Static password

It is hard for some users to use a new password for every transaction because they can not bear many passwords in their mind so that it is easier to use a static password. To provide the static password instead of the one-time password, we must consider that PassDigest is publicly propagated and it reveals the digest of the password which is the key to passing the step two of authentication. In fact, in the one-time password structure, we have two essential parts, PassVerifier and PassDigest, those parts are disposable and must not use for another transaction. As we can see in Fig.3, these parts are publicly propagated and revealed their data, therefore supposing we use the same parts and same public and private keys for another transaction, so from thief point of view, the thief would be able to use revealed PassDigest to pass the second step of authentication in the new transaction . As a result, it is mandatory to use a new password for every transaction.

To address this problem and develop the structure to be able to use static passwords, we need to have at least one character change in the PassDigest for every transaction in order to not revealed the digest of the password. For

doing this, we use salt [17]. for the hash algorithm. The salt is a predictable data which we add to our original data then use the new data as an input of hash algorithm. It makes our input different; as a result, we have a different PassDigest (8). We chose a name, sequence number (SN), as the salt. SN is a number which counts the number of output transactions from the first one when the public key and wallet address created and the first bitcoin was transferred to it until the current transaction as we see in Fig.4. Hence SN is predictable, and we can use it as the salt for creating PassDigest. It will be done by software, and user does not need to concern about and tries to count.

PassDigest=Hash("Password" + "SN") (8)

For example, Bob received some money by five transactions (input transactions) and spent some money by 3 transactions (output transactions); therefore SN is three as you can see in Fig.4, and we know, PassVerifier is depended on PassDigest and PassDigest is depended on SN; as a result Bob needs to regenerate his PassDigest and PassVerifier After every output transaction which a wallet or a software would do.

If Password is "12345" and SN=3

PassDigest3=Hash("12345"+"3") (9)

="77F919B0FFF753C0A6169C8ADFE2E7A570321D7009894D9D121BA77E2684F647"

PassVerifier3 =Hash(PassDigest3+PublicKey) (10)

At the moment, Bob needs to state PassVerifier3 (10) and his PublicKey for receiving transactions, and state PassDigest3 (9) with his signature created by his PrivateKey for sending.However, after another output transaction, Bob needs to state PassVerifier4 (12) and His PublicKey for receiving:

PassDigest4=Hash("Password" +"4") (11)

PassVerifier4 =Hash(PassDigest4+PublicKey) (12)

While the password is fixed, the PassVerifier and PassDigest will be changed, and it will protect security data from revealing. From a thief's point of view, the thief will not be able to use revealed PassDigest to pass the second step of verification because the data have changed, and for every transaction, we will have a new PassVerifier because it is derived from PassDigest which is depended on variable but predictable SN.

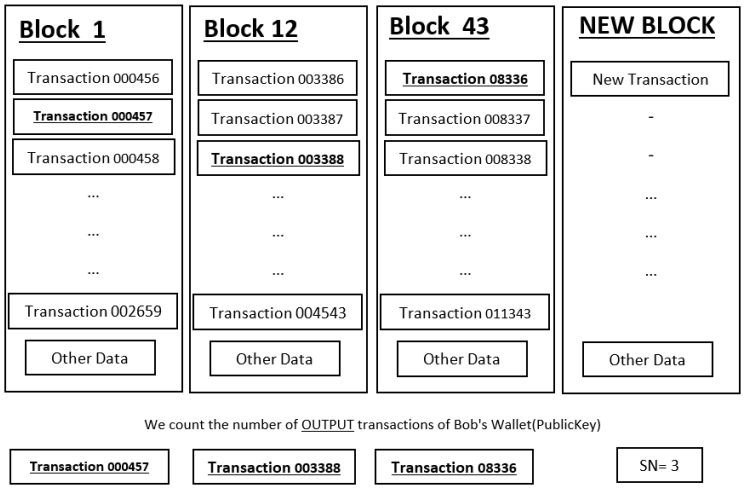

Figure 4: how we count sequence number

3.5 Security

From a thief's point of view, stealing bitcoins need to have the private key and the related password. Supposing, the thief somehow had access to the private key and now needs to pass the second step, which is the password verifier and perform transactions in his favor.Available Solutions for thieves are,first, direct access to the password related to the victim by spying him/her or obliging the user to tell the password which he/she generated it during or before receiving a transaction. It is evident if the user shares the password, he/she loses his/her assets, and it is not related to this article.Second, guessing a phrase which has the same output as the original password when the thief uses it as an input of the hash algorithm and uses the output to pass the second-stage of authentication, It means the thief would find a collision [18] . According to the analysis done by Henri GilbertHelena Handschuh [19], , and Bart Preneel[20], a good hash algorithm which has smallest output with excellent collision resistance is Secure Hash Algorithm 256 (SHA-256). In this algorithm, we try to perform a collision attack[18] and birthday attack [21] , If n is the number of algorithm output digits to find the collision,we have (13):

$$\text{Hash CyclesToFindTheCollision}=2^{n-1} \quad (13)$$

To find at least two phrases with the same output in the SHA-256 ,we need $2^{256}/2=2^{128}$ hash cycles. To do this, we need to access to one of the most powerful machines with the highest hash rate which is a distributed computing machine [22] namely Bitcoin Mining Machine. At the time of writing this article, it has the ability to 100 quadrillion hash per second (10^{17} H/S) [23], , if we can use this machine, we need $2^{128}/10^{17}\sim 3x10^{21}$ second ($\sim 10^{13}$ years) to find a collision; therefore, it is impossible. It is worth mentioning that the age of our earth is 4.5 billion years ($4.5x10^{9}$ years) [24], and the time of finding a collision is 10000 billion years ($10^{4}x10^{9}$ years).

4. Future Work

This structure needs to add some extra information to the transactions; hence, the transactions will have more data to store than before which would be a problem. As a result, we will need more storage to store transactions; therefore we will work on reducing the size of information mentioned (pass-verifier, pass-digest) which two-factor authentication needs them to be executed.

Another task is to implement the entire structure by using bitcoin script in the current bitcoin structure so that we can have this solution in regular bitcoin structure and we do not need to implement new blockchain network to have this solution.

The last part on which we will work is creating a distributed password recovery for this structure to complete it, and will work on creating a PassVerifier which would not need to change if the PassDigest changed.

5. Conclusion

We have shown how we can have an easy way to execute two-factor authentication with the help of hash algorithm which is practical and included all blockchain benefits because people used to utilize this structure in other financial methods such as conventional banking system. The user can use a memorable password and bears it in his/her mind,and it becomes a stage of authentication which is inaccessible to other people and without this step the transactions are not performed. So that the concern of keeping private key in a secure place will be reduced because It is no longer trouble and we can place a private key in regular locations because transactions will not be created if the correct password is not stated. Moreover, this structure needs minimal or conventional security to provide the highest security.Furthermore, as an additional option, any file can be replaced as a password. This file can be any

photo, music or text file. Finally, the concern about the security of using cryptocurrency will have vanished from people's mind because they will use the conventional security system which they used to rely on it.

Compliance with ethical standards

Conflict of interest Authors declare that they have no conflict of interest

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

[1] Debois S, Dumas M, Haarmann S, Jacobsen H-A, Jans M, Mendling J, et al., 2019, Two Perspectives on Blockchains: Capabilities vs. Features. Schloss Dagstuhl--Leibniz-Zentrum fuer Informatik.
[2] Zhang K, Jacobsen H-A. 2018, Towards Dependable, Scalable, and Pervasive Distributed Ledgers with Blockchains. ICDCS. p. 1337-46.
[3] Zhang K, Vitenberg R, Jacobsen H-A. 2018, Deconstructing Blockchains: Concepts, Systems, and Insights. DEBS. p. 187-90.
[4] Crosby M, Pattanayak P, Verma S, Kalyanaraman V, 2016, Blockchain technology: Beyond bitcoin, Applied Innovation, 2, 71.
[5] Singh S, Singh N. 2016, Blockchain: Future of financial and cyber security. 2016 2nd International Conference on Contemporary Computing and Informatics (IC3I), IEEE. p. 463-7.
[6] Nakamoto S, 2008, Bitcoin: A peer-to-peer electronic cash system.
[7] Wayner P. 1997. Digital cash: Commerce on the net: Academic Press Professional, Inc.
[8] Johnson D, Menezes A, Vanstone S, 2001, The elliptic curve digital signature algorithm (ECDSA), International Journal of Information Security, 1, 36-63.
[9] Paul E. 2017, What is Digital Signature-How it works, Benefits, Objectives, Concept. Opgehaald van emptrust: http://www. emptrust. com/blog/benefits-of ….
[10] Kaushal PK, Bagga A, Sobti R. 2017, Evolution of bitcoin and security risk in bitcoin wallets. 2017 International Conference on Computer, Communications and Electronics (Comptelix), IEEE. p. 172-7.
[11] Claessens J, Dem V, De Cock D, Preneel B, Vandewalle J, 2002, On the security of today's online electronic banking systems, Computers & Security, 21, 253-65.
[12] Douglass M. 2006, Currency dispensing ATM with RFID card reader. Google Patents.
[13] Franco P. 2014. Understanding bitcoin: Wiley Online Library.
[14] Mann C, Loebenberger D, 2017, Two-factor authentication for the Bitcoin protocol, International Journal of Information Security, 16, 213-26.
[15] https://en.bitcoin.it/wiki/Script
[16] MacKenzie P, Reiter MK. 2001, Two-party generation of DSA signatures. Annual International Cryptology Conference, Springer. p. 137-54.
[17] Gauravaram P. 2012, Security Analysis of salt|| password Hashes. 2012 International Conference on Advanced Computer Science Applications and Technologies (ACSAT), IEEE. p. 25-30.
[18] Wang X, Yin YL, Yu H. 2005, Finding collisions in the full SHA-1. Annual international cryptology conference, Springer. p. 17-36.

[19] Gilbert H, Handschuh H. 2003, Security analysis of SHA-256 and sisters. International workshop on selected areas in cryptography, Springer. p. 175-93.
[20] Preneel B. 2010, The first 30 years of cryptographic hash functions and the NIST SHA-3 competition. Cryptographers' track at the RSA conference, Springer. p. 1-14.
[21] Bellare M, Kohno T. 2004, Hash function balance and its impact on birthday attacks. International Conference on the Theory and Applications of Cryptographic Techniques, Springer. p. 401-18.
[22] Nurmi D, Brevik J, Wolski R. 2005, Modeling machine availability in enterprise and wide-area distributed computing environments. European Conference on Parallel Processing, Springer. p. 432-41.
[23] https://www.blockchain.com/en/charts/hash-rate
[24] Allegre CJ, Manhes G, Göpel C, 1995, The age of the Earth, Geochimica et Cosmochimica Acta, 59, 1445-56.

