SECURING CRYPTOCURRENCY WALLET SEED PHRASE DIGITALLY WITH BLIND KEY ENCRYPTION

Cheman Shaik

VISH Consulting Services Inc, 6242 N Hoyne Avenue, Chicago IL, USA

cheman_shaik@rediffmail.com

ABSTRACT

A cryptographic method of digitally securing cryptocurrency wallet seed phrase through Blind Key Encryption is discussed wherein two blind keys random in nature are generated and used to produce two ciphertexts. The mathematical algorithm used in blind key encryption is described in detail and also an explanation is provided as to how the encryption defeats hackers even after they could successfully compromise a ciphertext of the seed phrase along with its decryption key. Different scenarios of storing the ciphertexts are documented.

KEYWORDS

Cryptocurrency Wallet, Blockchain, Seed Phrase, Blind Key Encryption.

1. INTRODUCTION

Cryptocurrency is a digital currency that has recently gained lot of momentum in the financial sector. Unlike the fiat paper currency that we have been using, cryptocurrency leaves no trace of its owners and transactions. Cryptocurrency is mainly based on two things – block chain that serves as a ledger of the currency assets and transactions, and cryptography that serves as a means to protect the assets from adversaries.

While cryptocurrency has advantages such as low transaction fees, eliminated bank charges, anonymity of spenders, it also has disadvantages such as high price volatility, non-reversibility of transactions and vulnerability to attacks such as time jacking, double spending and selfish mining[1]. Another serious concern with cryptocurrency is the loss of assets when the seed phrase of the wallet managing the assets is lost or forgotten[2].

There are several crypto currencies currently traded in the market and continuously fluctuating according to their demand. Many cryptocurrency wallets are available on the internet that enable users to manage their crypto assets and perform transactions. When a client installs and sets up a wallet on his computer or mobile device, it generates a seed phrase of 12 - 24 words and mandates him to write down and store it in a secure place[3]. Seed phrase is also known as backup phrase and recovery phrase.

In fact, crypto assets are not stored in a cryptocurrency wallet but actually on a distributed, decentralized digital ledger called block chain on public nodes[4]. A cryptocurrency wallet provides only a doorway to the assets on the blockchain and will be able to identify them based on the Seed Phrase supplied by the user and fetch them into another wallet that he installs on a different device.
A seed phrase is generated as a series of 12 – 24 words picked from a list of 2048 easily memorable dictionary words specified by the BIP39 standard\textsuperscript{5}. The seed phrase is sensitive to the order of the words that constitute it. The words are picked according a random number generated on the user’s computer or mobile device.

Seed phrase is a very important piece of text that needs to be supplied by the crypto assets’ owner when his computer or laptop is lost, stolen, broken or corrupted. The wallet will recover his assets from the blockchain using his seed phrase. A seed phrase stored, even in encrypted form, on a server or a local computer or mobile device connected to the internet is susceptible hacks. If the seed phrase is compromised by hackers, they can steal all the assets by using it in a new wallet installed on their computer.

Cryptocurrency assets and their transaction are not traceable and therefore any assets lost are unrecoverable if the seed phrase is lost or forgotten. Cryptocurrency wallets strongly recommend that their users write down the seed phrase on a medium that is not exposed to the internet. However, a seed phrase stored on paper media is vulnerable to damages by fire, floods, and misplacement in shifting.

The vulnerability of seed phrase in both the aforementioned scenarios calls for a novel approach to its encryption. An encryption scheme that defeats hackers even when the seed phrase is stored on a medium exposed to the internet would address the issues of seed phrase in both these scenarios.

Any confidential information that needs to be secured from hacker is encrypted using different industry standard algorithms such as AES, RSA and ECC depending on the parties involved in communicating such information \textsuperscript{6}. In case the information is to be shared with a known party, a symmetric encryption algorithm such as AES is used to encrypt it, otherwise if it requires sharing between two unknown parties an asymmetric encryption algorithm such as RSA or ECC is used.

2. RELATED WORK

Most cryptocurrency wallets instruct their users to write down the seed phrase on paper and store multiple copies of the same in secured places \textsuperscript{2,7}.

Some metal devices such as BillFold are available in the market using which crypto asset owners can assemble their seed phrase by arranging some tiny metal tiles packaged with the product in sequence in slots of a metal frame\textsuperscript{8,9}. The frame can be preserved in a safe box for use when required in future.

Many wallets provide an option to encrypt seed phrase with a password which would serve as the 13\textsuperscript{th} or 25\textsuperscript{th} word of the seed phrase and will protect assets from hackers even in case of compromise \textsuperscript{10}.

Shamir’s Backup is a method used to secure seed phrase wherein the seed phrase is split into multiple unique shares. Recovering the wallet will require the user to collect and use a specific number of shares \textsuperscript{11}. However, Shamir’s Secret Sharing Scheme can be cheated when shares are revealed asynchronously and the cheater is the last one to reveal his share \textsuperscript{12}.

No evidences are found in literature on Blind Encryption of seed phrase that enables crypto wallet users to store the resulting ciphertext on devices connected to the internet and at the same time keep it secured from hackers.
3. **The Basic RSA Encryption Method**

The basic RSA encryption method works with a public key and a private key. Both these keys consist of a common value called the key modulus \(n\)\(^{13}\). The public key consists of another value called the public key exponent \(e\), and the private key a value called private key exponent \(d\). The basic mathematical equations used to create keys and encrypt and decrypt data are as follows:

**a. Key generation:**
\[ n = pq \] \[1\]
where \(p\) and \(q\) are two prime numbers

Euler Totient Function \(\phi = (p-1)(q-1)\) \[2\]
\[ ed = k\phi + 1 \] \[3\]
where \(e\) is the public key exponent, \(d\) is the private key exponent, and \(\phi\) is Euler Totient Function whereas \(k\) is any integer satisfying the above equation. All the four variables in the above equation must be integers.

The integer pair \((e, n)\), termed the public key, is published openly for use by the public for encryption of plain text messages.

The integer \(d\), termed private key, is very confidential and must be kept secret.

Once the public and private keys are generated, the Euler Totient Function \(\phi\) must be discarded leaving behind no trace thereof, which could otherwise be easily factored and lead to the computation of private key from the public key.

**b. Encryption and Decryption:**

Assuming \(M\) is a number representing a plain text message, a ciphertext \(C\) is computed by performing an exponential modular operation.
\[ C = M^e \mod n \] \[4\]
The ciphertext \(C\) is sent over the network to its intended recipient who will decrypt it to derive the original number \(M\) by performing another exponential modular operation as follows.
\[ M = C^d \mod n \] \[5\]
If a hacker gets access to the private key exponent \(d\), he can compromise all the messages encrypted by the public key \((e, n)\).

4. **Blind Key Encryption to Secure Seed Phrase**

Blind key encryption uses two randomized public key exponents to encrypt messages into two ciphertexts and two deterministic private key exponents to derive the original plain text. The two ciphertexts generated are stored on two different physically separated devices. In case a hacker manages to compromise one of the devices and gains access to one of the ciphertexts, it would
not be possible for him to decrypt it even if he has access to the entire private key, including both the decrypting exponents\(^{\text{[14]}}\).

The basic RSA key generation, encryption and decryption equations are enhanced for blind key encryption as follows:

**a. Key generation:**

\[
n = pq
\]  \[\text{[6]}\]

where \(p\) and \(q\) are two prime numbers

\[
\phi = (p-1)(q-1)
\]  \[\text{[7]}\]

\[
e_1d_1 + e_2d_2 = k_1\phi + 1
\]  \[\text{[8]}\]

\[
d_1 + d_2 = k_2\phi
\]  \[\text{[9]}\]

where \(e_1, d_1, e_2, d_2, k_1\) and \(k_2\) are all integers.

\((e_1, e_2, n)\) is used for encryption and \((d_1, d_2, n)\) is used for decryption.

**b. Encryption and Decryption:**

Assuming \(S\) is a number representing the seed phrase of a cryptocurrency wallet, two ciphertexts \(C_1\) and \(C_2\) are computed by performing exponential modular operations:

\[
C_1 = S^{e_1 + r} \mod n
\]  \[\text{[10]}\]

\[
C_2 = S^{e_2 + r} \mod n
\]  \[\text{[11]}\]

where \(r\) is a random integer in the range to \(0 < r < n\).

The ciphertexts \(C_1\) and \(C_2\) are stored on two physically separated computers, servers, mobile devices or any other storage media.

Whenever the wallet user wants to get the seed phrase, he can access the two ciphertexts \(C_1\) and \(C_2\) and decrypt them into the original seed phrase \(S\) using the below equation:

\[
S = [(C_1^{d_1} \mod n) (C_2^{d_2} \mod n)] \mod n
\]  \[\text{[12]}\]

**c. Proving \(S\) is the original seed phrase**

Substituting the expressions for \(C_1\) and \(C_2\) from equations 10 and 11,

\[
S = [(S^{e_1+r} \mod n)^{d_1} \mod n (S^{e_2+r} \mod n)^{d_2} \mod n] \mod n
\]  \[\text{[13]}\]

Simplifying the above,

\[
S = [(S^{eld_1 + rd_1} \mod n) (S^{e_2d_2 + r d_2} \mod n)] \mod n
\]  \[\text{[14]}\]
Further simplifying,
\[ S = S^{e_1 d_1 + e_2 d_2 + r d_1 + r d_2} \mod n \] \[ \text{[15]} \]

Taking out \( r \) as common,
\[ S = S^{e_1 d_1 + e_2 d_2 + r(d_1 + d_2)} \mod n \] \[ \text{[16]} \]

Substituting \( k_1 \phi + 1 \) for \( e_1 d_1 + e_2 d_2 \) and \( k_2 \phi \) for \( d_1 + d_2 \) from equations 8 and 9,
\[ S = S^{k_1 \phi + 1 + k_2 \phi} \mod n \] \[ \text{[17]} \]

Rewriting the above,
\[ S = S^{(k_1 + k_2) \phi + 1} \mod n \] \[ \text{[18]} \]

According to Euler’s theorem in number theory \( S^{k \phi + 1} \mod n = S \) for any \( S, k \) n where \( S \) and \( n \) are relatively prime. Hence \( S = S \).

5. **Mathematical Steps for Implementation**

The mathematical steps to compute keys, encryption and decryption are enlisted below:

a. **Key generation:**
   - Select any two prime numbers \( p \) and \( q \)
   - Compute \( n = pq \)
   - Compute \( \phi = (p - 1)(q - 1) \)
   - Select any integer \( d_1 \)
   - Select any random integer \( k_2 \) and compute \( d_2 = k_2 \phi - d_1 \)
   - Compute \( e \) and \( d_2 \) such that \( ed_2 = k \phi + 1 \) by running Euclidean algorithm
   - Select any integer \( e_2 \)
   - Compute \( e_1 = e + e_2 \)
   - Discard \( \phi \)

Use \((e_1, e_2, n)\) as public key and \((d_1, d_2, n)\) as private key

Euclidean algorithm is a proven set of steps to compute \( e \) and \( d \) values as integers satisfying the RSA key equation for a Euler Totient Function \( \phi \).

b. **Encrypt of Seed Phrase**
   - Convert the seed phrase to a number \( S \)
   - Select a random number \( r \)
   - Compute ciphertext \( C_1 \) and \( C_2 \) as below:
     \[ C_1 = S^{e_1 + r} \mod n \]
     \[ C_2 = S^{e_2 + r} \mod n \]
   - Store \( C_1 \) and \( C_2 \) on two physically separated storage media.
c. **Decryption of Seed Phrase**

- Get the two ciphertexts $C_1$ and $C_2$
- Compute $S_1 = C_1^{d_1} \mod n$
- Compute $S_2 = C_2^{d_2} \mod n$
- Compute $S = S_1 \cdot S_2 \mod n$

6. **Practical Implementation**

The practical implementation of Blind Key Encryption requires two physically separated storage media on protected servers, personal computers, mobile devices, reliable email accounts, storage accounts etc. Alternatively different combinations of the aforementioned media can be used.

Fig. 1 below describes the steps of encryption, storage, retrieval from storage and decryption of the seed phrase in a cryptocurrency wallet.

![Fig. 1 Blind key Encryption and Decryption of seed phrase in Cryptocurrency Wallet](image)

When the cryptocurrency wallet generates a seed phrase at the time installation and setup on a personal device, it should technically generate the public and private keys using the algorithm discussed in the previous sections. Further, it should generate a random integer, create two blind encrypting key exponents by adding the random integer to the original encrypting exponents, generate two blind ciphertexts using the blind key exponents, and pass them to two physically separated media of the user’s choice for storage.

a. **Manual Storage:**

The cryptocurrency wallet user can manually copy and store the two blind ciphertexts with their respective decrypting keys on any two storage media of his choice. When the user installs the wallet on a different device, he can copy the two ciphertexts and their respective decrypting keys and enter them in a screen of a newly installed wallet which would recompute the seed phrase and subsequently fetch his assets from the blockchain.
b. Automated Storage:

As an alternative to manual persistence, the wallet may submit the two ciphertexts with their respective decrypting keys to two REST endpoints on two different hosting servers maintained by two different parties that the user has configured during installation and setup of the wallet. The Restful endpoints must be protected by authentication credentials which need to be entered in a screen of the wallet in order to enable the wallet to successfully store the ciphertexts. Similarly, when the user installs the wallet on a different personal device, he will be required to enter the two REST endpoints and their respective authentication credentials so the wallet application can fetch the cipher texts and decryption keys in order to recompute the seed phrase.

c. Storing on Trusted Email Accounts

In another alternative, the two ciphertexts and their respective decrypting key components can be safely stored in two different trusted emails accounts such as Gmail and Hotmail which are highly reliable and recoverable even in case of a server crash due to their disaster recovery, continuity and replication facilities.

Fig. 2 below describes generating and storing the two ciphertexts of the seed phrase on two different trusted, reliable email accounts.

7. Security of Seed Phrase from Hackers

In case a hacker manages to compromise one of the webservers where a ciphertext and its corresponding decryption key component is stored, he will not be in a position to recompute the seed phrase because the hacker needs both the ciphertexts and decrypting key exponents. Successful decryption and reconstruction of the seed phrase is practically not possible since the hacker has no idea where the other ciphertext and its decrypting key component is hosted. Usually, hackers infiltrate such servers that have a weakness in network, firewall, operating
system and security software. Chances are very slim that a hacker identifies and compromises both the servers that are logically related to a particular seed phrase.

a. **Defeating Brute Force Attacks**

Decrypting a single ciphertext with its corresponding decrypting key exponent will always result into garbage as the ciphertext is generated using a random encrypting exponent. The randomness introduced in the ciphertext can be eliminated only by performing a combined exponential modular operation on both the ciphertexts.

In order to be successful in seed phrase computation with only a single ciphertext and its corresponding decrypting key exponent, the hacker needs to launch a brute force attack on the random value \( r \) which is the same order as the key modulus \( n \). Such an attack will result in as many decryptions as the order of the key modulus \( n \). The hacker needs to read all the resulting seed phrases and pick the right one, which is practically impossible.

b. **Defeating Factoring Attacks:**

Blind key encrypted ciphertext is immune to factoring attacks unlike the deterministic key encrypted ciphertext. When an attacker is in possession of a ciphertext resulting from the basic RSA encryption and its public key \((e, n)\), the highest computational effort that he needs exert is to factor the key modulus \( n \) into its factors \( p \) and \( q \). Using the two factors the attacker can compute the Euler Totient Function \( \phi = (p-1)(q-1) \) and subsequently derive the decrypting exponent \( d \) by running Euclidean algorithm on \( e \) and \( \phi \).

On the other hand, when a hacker is able to gain access to only of the two ciphertexts generated by the Blind Key Encryption along with the encrypting key \((e_1, e_2, n)\), he cannot crack the ciphertext even after successful factorization and computation of the private key exponents \((d_1, d_2)\), unless he is able to identify the storage location of the second ciphertext and compromise it. The randomness introduced in computing the first ciphertext defeats factorization attacks, thereby making it very suitable for securing the seed phrase.

8. **ECC VS RSA VS AES FOR SEED PHRASE ENCRYPTION**

Blockchain technology uses ECC encryption for encryption due to its smaller keys compared to the RSA keys for the same level of security required. However, as seed phrase is not frequently accessed but only rarely in critical situations such as loss of the computer or mobile device that the cryptocurrency wallet is installed on, RSA encryption can very well be used for encryption of the seed phrase.

AES is a symmetric key encryption scheme that uses the same key for encryption and decryption. It is best suitable for encrypting secret information that is not shared with any unknown party. It also runs quite faster compared to both ECC and RSA. However, it does not allow to generate random keys and random ciphertexts. On the other hand, Blind Key Encryption allows crypto asset owners to store the seed phrase on storage media exposed to the internet and at the same time protects it from hacks.
9. CONCLUSION

Cryptocurrency investors and traders require wallets to manage their crypto coins and perform transactions. These wallets generate a Seed Phrase formed with 12 or 24 words randomly selected from the BIP39 standard list of 2048 words, which the user is instructed to store it securely. The Seed Phrase will be required when a user’s laptop or mobile device that the wallet is downloaded on is lost, stolen, corrupted or destroyed. The user can recover his crypto assets by entering the same seed phrase in a new wallet on his new device.

Cryptocurrency wallet users have concerns with storage of the seed phrase. It is very hard to remember such a long seed phrase of 12 or 24 words. A seed phrase stored on paper media can be lost in fire and floods or damaged by humidity or termites. On the other hand, a seed phrase stored on a computer hardware, mobile phone, or server connected to the internet is vulnerable to hacks by attackers.

The seed phrase can be stored securely on servers, email accounts, and user’s personal devices through Blind Key Encryption that defeats hackers. Blind Key Encryption is an RSA based public key encryption method that uses two encrypting key components and two decryption key components. It generates two ciphertexts that are random in nature, each of which individually prevent the successful decryption and stealing of seed phrase, thereby saving the assets of the cryptocurrency wallet user. It requires both the ciphertexts of the seed phrase which are available only to the actual owner of the crypto assets.

The Blind Key generation algorithm generates two encryption exponents and two decryption exponents with a common key modulus using an enhanced RSA algorithm. Two blind encryption exponents are generated by adding a random to each of the two encrypting exponents. Two blind ciphertexts are generated by encrypting the original seed phrase with the two blind encryption exponents. The two ciphertexts are stored on two different hosted servers, email accounts or personal devices. In case any of the two ciphertexts is compromised in a hacking incident, the hacker will be unable to decrypt it to the original seed phrase as both ciphertexts will be necessary to derive the original seed phrase.

The encryption scheme eliminates the need to store the seed phrase on paper media. It enables crypto asset owners to encrypt their seed phrase and store it on digital media, and at the same time protect it from hackers.

In regards to performance of the Blind Key Encryption, it takes twice the time taken by the regular RSA encryption method as it needs to compute two ciphertexts. Also, when compared to ECC encryption, RSA encryption is bit slower. However, the security advantage that the Blind Key Encryption offers to seed phrase which is crucial to the security of crypto assets outweighs its slow performance, because security prevails over rest all other aspects, especially in case of cryptocurrencies.

A recommendation for future work is to work on the ECC encryption algorithm and enhance it to work like the RSA Blind Key Encryption algorithm wherein randomized keys can be generated and two or more ciphertexts can be computed which will yield the original text on decryption.
REFERENCES


AUTHOR

Cheman Shaik is a Research & Development professional in Computer Science and Information Technology for the last twenty years. He has been an inventor in these areas of technology with eight U.S Patents for his inventions in Cryptography, Password Security, Codeless Dynamic Websites, Text Generation in Foreign Languages, Anti-phishing Techniques and 3D Mouse for Computers. He is the pioneer of Absolute Public Key Cryptography in 1999. He is well known for his Password Self Encryption Method which has earned him three U.S Patents. He has published research papers in IJCSEA and the proceedings of EC2ND 2006 and CSC 2008.