Wu *et al. Journal of Cloud Computing* (2023) 12:39
https://doi.org/10.1186/s13677-023-00417-7

Journal of Cloud Computing:
Advances, Systems and Applications

RESEARCH

Open Access

Cloud-edge data encryption in the internet of vehicles using Zeckendorf representation

Yun Wu[1], Liangshun Wu[2] and Hengjin Cai[1*]

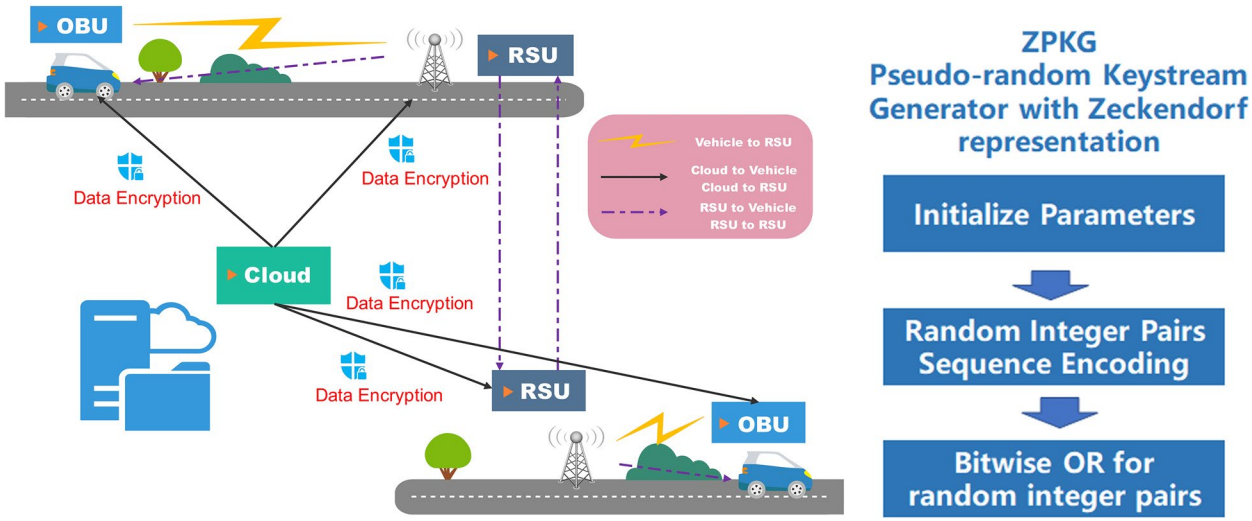
Abstract

Cloud-edge data security is a key issue in the internet of vehicles (IoV), as the potential for data breaches increases as more vehicles are connected. As vehicles become smarter and more connected, the risk of unauthorized access to the data generated by the vehicles also increases. Data encryption is a highly effective security measure that is widely used to protect the IoV from malicious actors. By encrypting data, it becomes virtually impossible for unauthorized individuals to access the information. This ensures that only the intended parties can access the data, allowing for secure communication between cloud and edge. Data encryption is a cost-effective and reliable security measure that is essential for any organization that relies on the IoV. The IoV is characterized by the large volume of data that is exchanged between devices in cloud and edge. This necessitates the use of a strong encryption method, such as stream ciphering, which is particularly well-suited to this type of environment. Stream ciphering provides the highest levels of security, making it the ideal choice for securing data transmission in the IoV. Many stream ciphering algorithms use bitwise exclusive or (XOR) to encrypt the data stream, so the core is the generation of a pseudo-random key stream. This paper proves that the probability of the number 1 appearing in the middle part of the Zeckendorf representation is constant, which can be used to generate pseudo-random key stream sequences. The pseudo-random sequence generated by the linear feedback shift register (LSFR) is periodic, and the key sequence will be duplicated. The logistic chaos (LC) sequence is too sensitive to the disturbance of initial value, and its stability is poor. In this paper, our proposed ZPKG (key generator based on Zeckendorf presentation) algorithm solves these two main problems in stream ciphering. The generated key sequence not only has strong randomness, but also is infinitely long, and it is robust to the minor disturbance of the initial value.

Keywords Internet of vehicles, Cloud-edge communication, Data encryption, Zeckendorf representation

*Correspondence:
Hengjin Cai
hjcai@whu.edu.cn
Full list of author information is available at the end of the article

Graphical Abstract



Introduction

Internet of Vehicles (IoV) is an emerging technology that enables the connection of vehicles to the Internet, allowing them to communicate with other connected devices. This technology enables vehicles to be monitored and controlled remotely, allowing for advanced safety features such as automatic braking and lane departure warnings. It also allows for improved fuel efficiency and emissions management and the integration of other digital services such as navigation, entertainment, and more. IoV is a rapidly developing technology with the potential to revolutionize the automotive industry, making vehicles safer and more efficient. Big data analytics can be used to gain valuable insights from the data collected, allowing for better decision-making and predictive capabilities. With the ever-increasing sophistication of technology, the sensors can detect a multitude of variables, from the vehicle's speed and direction to the engine's temperature and fuel level. Figure 1 gives examples of the data transmitted in the IoV, including advisory speed, flags, vehicle speed, driver input, raw position, lane level position, and so on. The data are often transmitted to cloud servers for Map Matching, Digital Twin Visualization, Human Behavior Modeling, Motion Planning, and Control, creating insights into the vehicle's performance and making recommendations for better driving and maintenance. By combining the power of IoV and Big Data, businesses can make better decisions and create innovative solutions which can improve the lives of individuals and communities.

A typical model of IoV is comprised of three parties:

- Cloud: The Cloud is considered reliable and trustworthy. Periodically, the RSU sends data packets to the Cloud. The data packets contain critical information such as car conditions, real-time traffic conditions, etc. The Cloud records the properties of each car and broadcast traffic statistics.
- RSUs: RSUs (Road Side Units) are roadside units in the ETC system that are supported by Dedicated Short Range Communication (DSRC) technology. They interact with OBUs. RSUs are utilized for vehicle identification as well as computerized point deduction. RSUs are often put on the side of the road and unattended express lanes, which are responsible for the management of highways and vehicle yards.
- OBUs: OBUs (Short for On Board Units) is a microwave device that communicates with RSUs through Dedicated Short Range Communication (DSRC) technology. The OBUs on the vehicle and the RSU-Road Side Units on the side of the road interact with each other through microwaves in the ETC system. When the vehicle speeds through the RSU, the OBU, and the RSU interact with microwaves, much like our contactless card, but from a distance of a dozen meters and at a higher frequency of 5.8GHz. When it passes, it determines the authenticity, receives the model, computes the rate, and subtracts the toll.

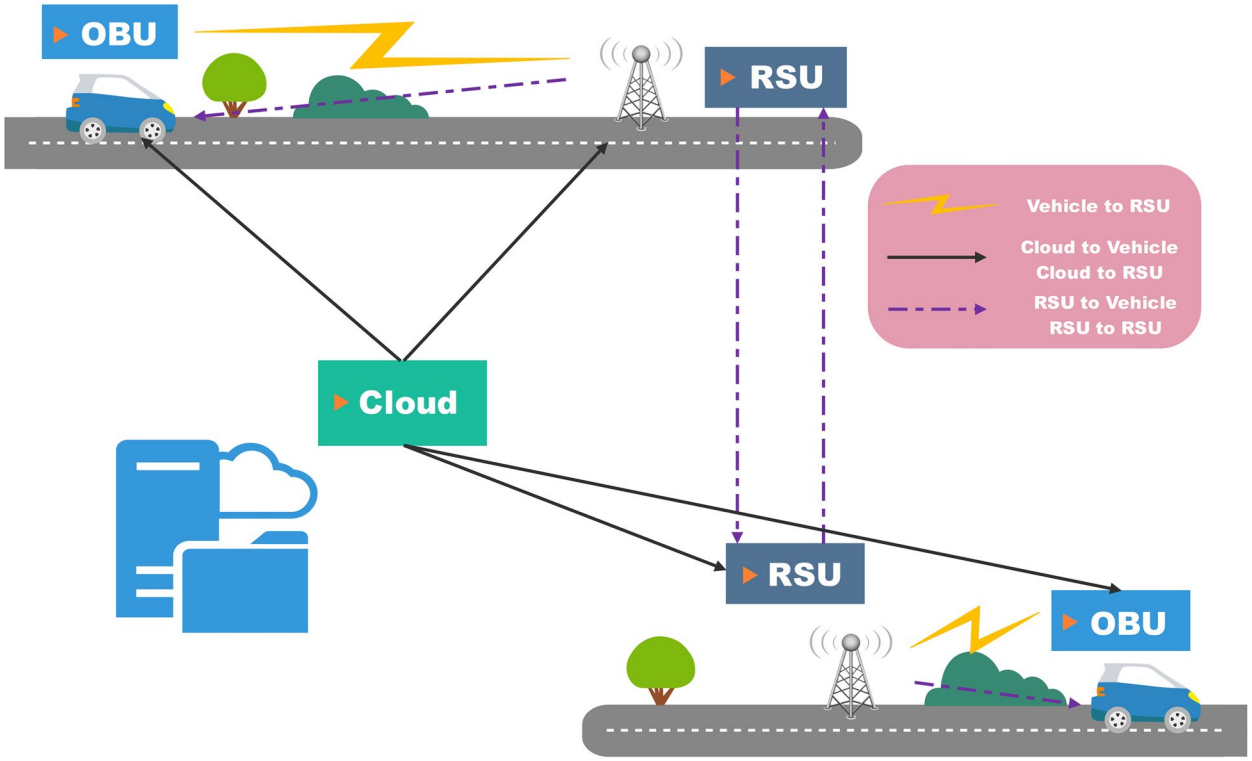


Fig. 1 The data transmitted in IoV

Figure 2 depicts cloud-edge communication scenarios, where OBUs and RSUs act as edge devices and they communicate with the cloud.

Cloud-edge data security is a critical issue in IoV, as this technology introduces a range of new cybersecurity risks. Automated vehicles are connected to the cloud for critical services such as navigation, safety and security, and fleet management. However, this also means that the data and communication generated by these vehicles are vulnerable to cyber-attacks. The data could be intercepted and manipulated, leading to potential safety issues, or the vehicle itself could be taken over

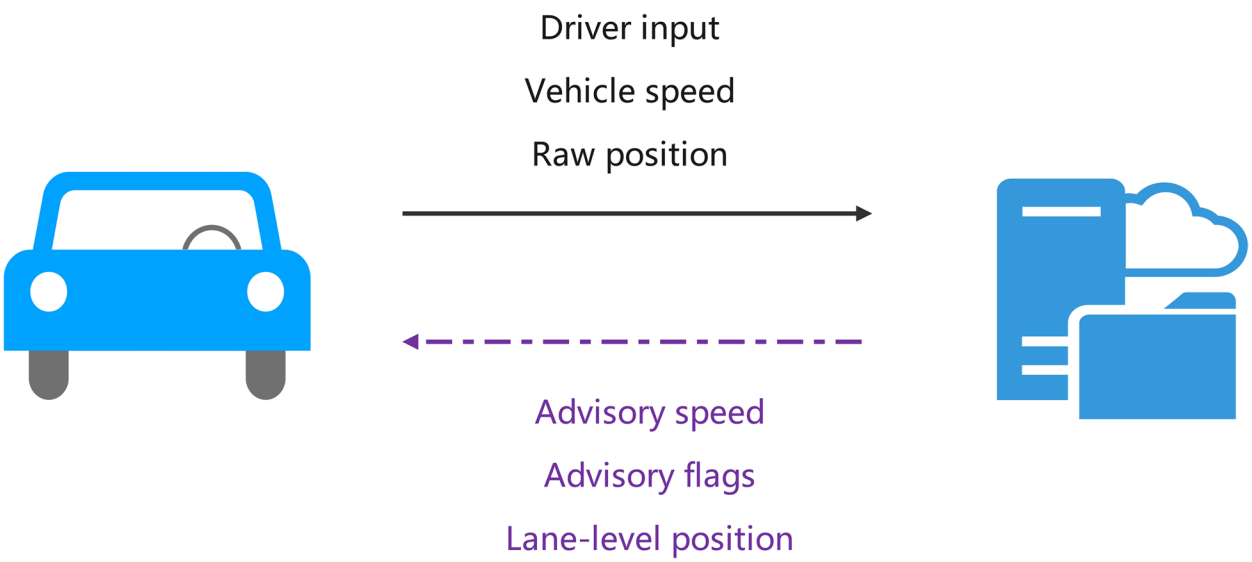


Fig. 2 Cloud-Edge (RSU and OBU) communication scenarios in IoV

by malicious actors. Data encryption is one of the most powerful tools available to address these security concerns, which offers an effective way to protect sensitive information from unauthorized access. By scrambling data and making it unreadable, encryption provides an extra layer of security to help ensure the privacy and integrity of data. The IoV has the feature of big data.

Before the large-scale application of Quantum Cryptography, stream ciphering was the most suitable encryption method for "big data" [1]. For stream ciphers, pseudo-random sequence generation (PRNG) is very important. For normal applications, where good statistical properties and speed are important, consider O 'Neill's PCG collection, Vigna's xoroshiro family, such as xoroshiro128 + (not the Japanese name btw, but "X or, rotate, shift, rotate "), and D. E. Shaw's Random123 suite (including Philox and the well-named ARS, which is a simplified form of encrypting infinite zero sequences using AES-CTR), though I'm not sure how much checking has been done with Random123 PRNG. To encrypt secure PRNG(CSPRNG) For encryption applications where unpredictability is important, consider using cryptographically secure PRNGS, for example, Bernstein's ChaCha20, RF C7539. The alternative is Wu's HC-256 and Jenkins' ISAAC64. The downside of these algorithms is that they're too complicated. Linear Shift Register (LSFR) and Logistic chaos (LC) are the two popular PRNGs for stream ciphering. Logistic Chaos is a PRNG algorithm that utilizes chaotic dynamical systems to generate unpredictable numbers. LSFR is a PRNG algorithm that uses linear feedback to generate a sequence of pseudo-random numbers. Both algorithms have been used for various applications, including cryptographic applications, simulation, and game design. Both algorithms are capable of generating statistically random numbers, making them useful for applications that require a certain degree of randomness. However, the pseudo-random sequence generated by the linear feedback shift register (LSFR) is periodic, and the key sequence will be repeated. LC method, which is used most currently, however, has a synchronization problem. That is, it is too sensitive to the disturbance of initial parameters and has poor stability.

Fibonacci numbers have been known to mathematics for some 800 years and it seems rather surprising that this property of theirs did not receive attention until relatively recently. Indeed, nothing appeared in print concerning the Zeckendorf representation until the middle of the 20th century, with the publication of a paper by C.G. Lekkerkerker [2]. Zeckendorf did not publish his account until 1972 (see [3]) although he had proof of his theorem by 1939. Every positive integer can also be uniquely represented as a sum of the different powers of two, and it is natural to compare Zeckendorf representations with binary representations. This is pursued in [4], where there is a discussion of algorithms that, given two positive integers a and b in the Zeckendorf form, produce the Zeckendorf forms for a + b. Zeckendorf's Theorem points out that natural numbers are the sum of one or more different Fibonacci numbers [5]. In other words, a positive integer n can be expressed as: $N = F_{k_1} + F_{k_2} + F_L + F_{k_r}$. For example: $17 = F_1 + F_4 + F_7$, $20 = F_3 + F_5 + F_7$. If all Fibonacci numbers in the above decomposition are mapped to a bit string composed of 0,1 according to their positions, that is, let the positions $k_1, k_2, \ldots, k_r$ of the sequence be 1, and the rest positions are 0, a new coding system represented by 0,1 can be obtained. This coding method, also known as Zeckendorf-Lekkerkerker coding, is the Ostrowski coding system [6, 7]. In recent years, The research in this field is focused on the discussion of mathematical properties. Among other results, Bugeaud [8] establishes, in a quantitative form, that any sufficiently large integer cannot simultaneously be divisible only by very small primes and have very few digits in its Zeckendorf representation. Idziaszek [9] investigates a relationship between the numeral system using Zeckendorf representations and the golden ratio numeral system. Alecci et al. [10] determine the Zeckendorf representation of the multiplicative inverse of a modulo F_n. Vukusic et al. [11] find an explicit upper bound for y^a, which only depends on the Hamming weight of y concerning the Zeckendorf representation. Other influential work includes [12]. The Zeckendorf representation has many advantages, such as non-uniqueness of coding, strong regularity of probability structure, periodicity of modules, and anti-interference, which may provide a new idea for the breakthrough of data encryption in IoV.

In what follows, based on the probability structure properties of the Zeckendorf representation, we propose a pseudorandom keystream generator (PKG), which can be used to encrypt cloud-edge IoV data. The proposed PKG algorithm addresses the periodicity problem of linear feedback shift registers (LFSRs) as well as the synchronization problem of LC.

The following content is organized as follows: Related work section reviews the literature, The proposed algorithm section proposes our algorithm, Security analysis section carries out security analysis theoretically, Experiments and discussion section conducts experiments and discusses the results, and Conclusion section concludes.

Related work

Modern cryptography comprises symmetric encryption, asymmetric encryption, and hash functions, with symmetric encryption further split into block ciphers and stream ciphers. Block ciphers accept fixed-size plaintext as input, such as 64-bit, 128-bit, 256-bit, and so on.

Permutation and diffusion are also used. However, when large volumes of data with high real-time needs are sent, block encryption algorithms such as DES and AES fail to meet the requirements due to their strong correlation and high redundancy [13].

Stream ciphers are lightweight encryption algorithms that are ideal for big data. Stream ciphers are considerably faster to perform than block ciphers. Stream ciphers are classified into two types: those based on bytes and those based on pseudo-random number sequences, with the latter being more extensively employed in practice. RC4 is a byte-based stream ciphering method that uses the PRGA algorithm to create pseudo-random numbers and then uses the KSA encryption technique to accomplish particular encryption operations. RC4 is widely used in SSL/TLS protocol and WEP protocol as an IEEE802.11 WLAN standard [14]. The key of a stream ciphering technique based on a pseudo-random number sequence changes at random and every bit of data is encrypted with every bit of key (typically XOR / XOR operation), making cracking theoretically impossible (one-time encryption principle). Because the XOR technique is simple to implement in hardware, it allows for real-time encryption of high-speed large data. The statistical inspection institution has set stringent random criteria for the pseudo-random number sequence. Berger et al. [15] developed a new pseudo-random number generator that is resistant to common assaults and can increase internal diffusion properties greatly. In recent years, there are many pieces of research on big data encryption. For example, Hosseini [16] examines the fingerprint system, classifies different aspects of it, such as techniques for extracting features and matching procedures, and identifies various vulnerabilities. He then proposes three mitigation strategies, including software (big data and encryption/ decryption) and secure hardware techniques. Zhou et al. [17] present a chaotic secure communication strategy based on the synchronization of multiple complex dynamical networks with double layers. Djamaluddin et al. [18]'s goal is to verify a real-time data transport and analytics environment in terms of encryption and access control. Chen et al. [19] present a massive data encryption technique based on a de-redundancy approach.

This study is not the first to propose research on stream ciphers utilizing Zeckendorf representation. Carry register feedback (FCSR) is critical in the hardware design of a stream cipher. Lin et al. [20] refined the FCRS circuit and developed a homogeneous Zeckendorf representation-based quark hash function that employs FCRS to produce random integers [21]. However, this type of study focuses on the hardware level rather than the software level production of stream cipher key streams; also, these studies have not thoroughly examined the properties of the Zeckendorf representation. The canonical Zeckendorf representation is only used in the random number generator to raise the disorder degree of the sequence; it has not been investigated for the construction of randomness.

The proposed algorithm

Requirements for stream ciphers of big data

RC4 stream cipher technology is a word-based stream ciphering method with variable key length and a large number of permutation operations, making it more appropriate for software implementation. However, one out of every 256 keys in RC4 is a weak key susceptible to state table analysis attacks [22]. Furthermore, cryptanalysis can uncover a significant correlation between the key's bytes, and if the same key sequence is repeated, the hacker can break the ciphertext. If the first three words of the key are discovered, iteration may be used to retrieve each word of the key used in RC4 [23]. Unlike RC4, which encrypts by byte, the LC and LFSR use a pseudo-random key stream to encrypt. Linear feedback shift registers (LFSRs) are excellent for hardware implementation due to their ease of algebraic analysis and high unpredictability (they can be successfully built by simple logic gates and registers). The primary drawback of LFSR is that it is periodic, which means that it can only be used to encrypt particular length sequences. Second, because its sequence is not a simple monotone sequence, it must be initiated [24]. The most often used stream ciphering technology is chaotic encryption. The difficulty with encryption is that it is excessively sensitive to the initial values of parameters due to its nonlinearity and synchronization [25]. To address the drawbacks of the LC and LFSR, a new pseudo-random key stream generation method with the following features must be investigated: First, the key stream sequence is sufficiently random. Second, the key stream sequence is lengthy enough to encrypt a large quantity of data. Third, the key stream sequence should not be very dependent on the starting value.

(1) Randomness

XOR is commonly used in stream ciphers. If numerous data are encrypted with the same key string in this situation, the data may be decoded without knowing the key sequence. The following are the reasons: suppose P_1 and P_2 are two strings of plaintext data, K is the key used to encrypt the data. The ciphertext: $E_1 = P_1 \oplus K$, $E_2 = P_2 \oplus K$, $\oplus$ denotes XOR operation. Because $E_1 \oplus E_2 = P_1 \oplus K \oplus P_2 \oplus K = P_1 \oplus P_2 \oplus (K \oplus K) = P_1 \oplus P_2$, XOR P_2 on both sides, we obtain $E_1 \oplus E_2 \oplus P_2 = P_1 \oplus P_2 \oplus P_2 = P_1$. The ciphertext is cracked.

The Golomb randomness hypothesis [26] defines the basic characteristics that a sequence needs to have if it is considered random enough. Include:

C1: In a periodic segment, the difference between the numbers of 1 s and 0 s is no more than 1.

C2: In a periodic segment, if the length of runs accounts for the total number of runs, here it is assumed that there are at least two lengths of runs.

C3: Autocorrelation function:

$$R(\Delta)=\sum_{i=1}^{L} l_i l_{i+\Delta}=\begin{cases} L/2, \Delta=0 \\ L/4, 0<\Delta<L \end{cases} \tag{1}$$

L denotes the length of the sequence, Δ denotes the interval.

(2) Infinity

Because the LSFR pseudo-random sequence is periodic, key duplication will occur if the plaintext length exceeds the period of the key sequence. As a result, given a large key space, the key stream sequence must be both random and aperiodic. In theory, this needs an infinitely long key sequence.

(3) Stability

The LC method is extremely sensitive to the initial value, and it is precisely because of this sensitivity that it produces randomness, which is also the origin of the term "chaotic," that is, the phenomenon in which the dynamic system appears random but is not random; however, a minor change can result in different encryption results, implying that chaotic systems are highly uncontrollable.

To some extent, the key sequence must be sensitive to the starting value. If it is not sensitive, all sequences will be the same or equivalent, and the intended unpredictability will be lost. The key sequence, however, will be easily broken if the initial value sensitivity is too high.

Algorithm principle

Assuming that there are m ones in the Zeckendorf representation sequence, the coding sequence is marked as: $Z_{L,m}$, according to Filipponi and Wolfowicz's discovery in 1987 [27], the number of possible sequences is

$$N_{L,m}=\begin{cases} \binom{L-m+1}{m}, 0\le m\le \left\lceil \frac{L+1}{2} \right\rceil \\ 0, m>\left\lceil \frac{L+1}{2} \right\rceil \end{cases} \tag{2}$$

$N_{L,m}(k)$ indicates the number of all Zeckendorf representation sequences with k-th bit being 1. Filipponi et al. proved that:

$$N_{L,m}(k)=N_{L,m}(L-k+1) \tag{3}$$

and

$$N_{L,m}(k)=\sum_{i=0}^{k-1}(-1)^i\binom{L-m-i}{m-1-i}N_{L,m}(k) \tag{4}$$

Among them.or $1\le k\le \left\lceil \frac{L+1}{2} \right\rceil$.Or

$$N_{L,m}(k)=\frac{1-(-1)^k}{2}\binom{L-m-k+1}{m-k}+\sum_{i=0}^{\left\lceil \frac{k-2}{2} \right\rceil}\binom{L-m-2i-1}{m-2i-1} \tag{5}$$

Based on the following assumptions [28]

$$\binom{a}{-|b|}=0 \tag{6}$$

It can be inferred from (4) that $i>m-1$.

Theorem 1 *If $m<k<L-m+1$, $N_{L,m}(k)=N_{L,m}(m)$ is a constant.*

Proof From (4), we have

$$\begin{aligned} &N_{L,0}(k)=0 \\ &N_{L,1}(k)=1 \\ &N_{L,2}(k)=\begin{cases} L-2, k\in\{1,L\} \\ L-3, k\notin\{1,L\} \end{cases} \end{aligned} \tag{7}$$

From (5), we have

$$\begin{aligned} N_{L,m}(m)=&\frac{1-(-1)^m}{2}\binom{L-m-k+1}{m-k} \\ &+\sum_{i=0}^{\left\lceil \frac{m-2}{2} \right\rceil}\binom{L-m-2i-1}{m-2i-1} \end{aligned} \tag{8}$$

Then,

$$\begin{aligned} &N_{2m-1,m}(m)=\begin{cases} 1, 2\nmid m \\ 0, 2|m \end{cases} \\ &N_{2m,m}(m)=\left\lceil \frac{m+1}{2} \right\rceil \\ &N_{2m+1,m}(m)=\begin{cases} (m+1)^2/4, 2\nmid m \\ m(m+2)/4, 2|m \end{cases} \end{aligned} \tag{9}$$

Therefore,

$$\Pr_{L,m}(k)=N_{L,m}(k)/N_{L,m} \tag{10}$$

Similarly, using (7) and (9), we can get

$$\begin{aligned}
&\Pr_{L,m}(1) = \Pr_{L,m}(L) = \tfrac{m}{L-m-1} \\
&\Pr_{L,m}(2) = \Pr_{L,m}(L-1) = \tfrac{m(L-2m+1)}{(L-m-1)(L-m)} \\
&\Pr_{2m-1,m}(m) = \begin{cases} 1, 2 \nmid m \\ 0, 2 \mid m \end{cases} \\
&N_{2m,m} = \begin{cases} 1/2, 2 \nmid m \\ m/(2m+2), 2 \mid m \end{cases} \\
&N_{2m+1,m} = \begin{cases} (m+1)/(2m+4), 2 \nmid m \\ m/(2m+2), 2 \mid m \end{cases}
\end{aligned} \tag{11}$$

Algorithm implementation

Based on the characteristics of the Zeckendorf representation structure, a pseudo-random key stream generation algorithm is designed.

Suppose there is a pair of keys (e_1, e_2), which are calculated as follows:

$$\begin{cases} e_1 = (a_n, c_n, \mu_n) \\ e_2 = (a_m, b_m, \mu_m) \end{cases} \tag{12}$$

where μ_n, μ_m are two prime numbers of the same order of magnitude. There is a pseudo-random sequence of integers

$$\begin{cases} \mathbf{N} = (N_1, N_2, \cdots, N_H) \\ \mathbf{M} = (M_1, M_2, \cdots, M_H) \end{cases} \tag{13}$$

Its initial value satisfies:

$$\begin{aligned}
&N_0 > 0, a_n, c_n < \mu_n \\
&M_0 > 0, a_m, c_m < \mu_m \\
&N_0 \neq M_0
\end{aligned} \tag{14}$$

Each of **M** and **N** is

$$\begin{cases} N_{h+1} = (a_n N_h + c_n) \bmod \mu_n \\ M_{h+1} = (a_m M_h + c_m) \bmod \mu_m \end{cases}, 0 \le h \le H-1 \tag{15}$$

μ_n, μ_m are prime numbers that satisfy $N_{h+1} \neq M_{h+1}$ 。 H is determined by the message length. Then

$$\begin{cases} Z_L(N_i) = (n_1, n_2, \cdots, n_L) \\ Z_L(M_i) = (m_1, m_2, \cdots, m_L) \end{cases} \tag{16}$$

Next, perform OR operation on the middle part (the part with constant probability) to get C_i of length t:

$$C_i = \{c_1, c_2, \cdots, c_t\} \tag{17}$$

where,

$$t = L - 2U_L + 2 \tag{18}$$

and

$$U_L = \frac{5(L+2) - 8 - \left[5(L+2)^2 + 4\right]^{1/2}}{10} + 1 \tag{19}$$

$$c_j = n_k + m_k, j = 1, 2, \cdots, t, k = U_L + j - 1 \tag{20}$$

Concatenate C_i, we obtain C of length Ht.

$$C = \{C_1, C_2, \cdots, C_H\} \tag{21}$$

C is the desired pseudo-random key stream. We call this pseudo-random key stream generation algorithm "ZPKG" (key generator based on Zeckendorf presentation).

Security analysis

Randomness analysis

Let F_r be the maximum Fibonacci number no larger than N_i, then the shortest Zeckendorf sequence is $S_L(N_i)$ with length $L = n - 1$. We can prove

$$L = \left\lfloor \log_\Phi \sqrt{5}\left(N_i + \frac{1}{2}\right) \right\rfloor - 1 \tag{22}$$

where $\Phi = \frac{1+\sqrt{5}}{2}$ denotes the Golden ratio.

$$p(1) = \Pr_{L,U_L}(U_L) = N_{L,U_L}(U_L)/N_{L,U_L} \tag{23}$$

When L goes to infinity, $\Pr_{L,U_L}(U_L)$ converges. In this case, $L > 25$. Then, the k-th number of $Z_L(N_i)$ and $Z_L(M_i)$ is

$$\Pr(0) = p^2(0) \approx \Phi^2/5 \approx 0.524 \tag{24}$$

Among them,

$$p(0) = 1 - p(1) \approx (\Phi + 1)/(\Phi + 2) \approx 0.724 \tag{25}$$

Similarly,

$$\Pr(1) = 1 - \Pr(0) = 1 - p^2(0) \approx 0.476 \tag{26}$$

From (24) and (26), it can be known when $L > 25$ Golomb's first randomness hypothesis is satisfied. Golomb's second randomness hypothesis is difficult to prove directly. We can verify whether Pr(00), Pr(01), Pr(10) and Pr(11) are satisfied. As per the definition of Zeckendorf representation, there is no 11 pair. Therefore, we compute

$$p(01) = p(10) = p(1) \approx 1/(\Phi + 2) \tag{27}$$

and

$$p(00) = 1 - p(01) - p(10) = 1 - 2p(1) \approx \Phi/(\Phi + 2) \tag{28}$$

Perform logical OR operation to get all pairs.

$$
\begin{aligned}
&(00) = (00) + (00);\\
&(01) = (01) + (00), (00) + (01) \text{ or } (01) + (01);\\
&(10) = (10) + (00), (00) + (10) \text{ or } (10) + (10);\\
&(11) = (10) + (01) \text{ or } (01) + (10)\circ
\end{aligned}
$$

From (27) and (28), we have

$$
\begin{aligned}
&\Pr(00) = p^2(00) = 1/5 = 0.2\\
&\Pr(01) = \Pr(10) = 2p(00)p(01) + p^2(01) \approx \Phi/5 \approx 0.324\\
&\Pr(11) = 2p^2(01) \approx 2/\left(5\Phi^2\right) \approx 0.152
\end{aligned} \tag{29}
$$

Golomb's second randomness hypothesis is thus satisfied.

Infinity analysis

When $H \to +\infty$, **N** and **M** goes to infinity.

$$
\begin{cases} \mathbf{N} = (N_1, N_2, \cdots, N_H) \\ \mathbf{M} = (M_1, M_2, \cdots, M_H) \end{cases}, H \to +\infty \tag{30}
$$

In this way, the pseudo-random key stream that is obtained by OR operation with the middle part of the Zeckendorf representation of the integer pair tends to be infinitely long, that is

$$
C = \{C_1, C_2, \cdots, C_H\} \to \infty, \text{if H} \to +\infty \tag{31}
$$

That is, the sequence length of C is infinite.

Stability analysis

According to the definition of each item of **N** and **M**

$$
\begin{cases} N_{h+1} = (a_n N_h + c_n) \bmod \mu_n \\ M_{h+1} = (a_m M_h + c_m) \bmod \mu_m \end{cases}, 0 \le h \le H - 1 \tag{32}
$$

Where $e_1 = (a_n, c_n, \mu_n)$, $e_2 = (a_m, b_m, \mu_m)$ are the initial parameters. We make a small change in the initial parameter, for example, make c_n be $c_n = c_n + 1$, then

$$
N_{h+1} = (a_n N_h + c_n + 1) \bmod \mu_n \approx N_{h+1} + 1 \tag{33}
$$

If encode it as Zeckendorf representation again, then

$$
\begin{aligned}
&Z\left(N_{h+1}\right) = 1, 0, 1, \cdots, 0_{F_1}, 0\\
&Z\left(N_{h+1}\right) = 1, 0, 1, \cdots, 1_{F_1}, 0
\end{aligned} \tag{34}
$$

The new sequence only changes from 0 to 1 at F_1, while the position of the sequence segment that generates the pseudo-random key stream is: $[m, L-m+1]$. That is to say, the generated pseudo-random key stream will not change.

Experiments and discussion

Statistical properties test

Using the FIPS140–2 [29] standard to test the random performance of the output keystream sequence. The four main tests are as follows:

(1) Monobit test: The number M of "1" in the output 20,000bit stream sequence satisfies $9725 < M < 10275$.
(2) Poker test: Divide 20,000-bit stream sequences into 5000 continuous 4-bit blocks, and the following conditions are required $f(i)$ is the number of decimal values i of 4-bit blocks, where $0 < i < 15$, the test value $X = \frac{16}{5000}\sum_{i=0}^{15} f^2(i) - 5000$, passes the standard $2.16 < X < 46.17$.
(3) Run test: The number of continuous "1" or "0" in the 20,000-bit stream sequence is required to meet the standard in Table 1.
(4) Long run test: It is required that the length of continuous "1" in the 20,000-bit stream sequence cannot exceed 26.

To test the randomness of the keystream sequence generated by the algorithm, each of the four main tests was performed k = 200 times, each time using a different initial key. The results are shown in Figs. 3 and 4.

Scrambling effect

(1) First check the relevance of the key. The key autocorrelation function obtained by randomly generating a 2048-bit key stream is shown in Fig. 5a. The autocorrelation function of the ciphertext obtained by encrypting the same plaintext "vehicle-related data...traffic-related data" using these keys is shown in Fig. 5b.

It can be seen that the correlation function of the key and the ciphertext has no repetition period, has characteristics similar to white noise, and has good autocorrelation.

(2) Information entropy is one of the measurement indicators of uncertainty. The ciphertext (2048bit) information entropy is shown in Table 2.

According to information theory, when the probability of occurrence of each symbol of the sequence (in binary terms, 0 or 1) is equal, the informa-

Table 1 Run test pass standard

Length of continuous "1"	Number
1	2315 ~ 2685
2	1114 ~ 1386
3	524 ~ 723
4	240 ~ 384
5	103 ~ 209
6+	103 ~ 209

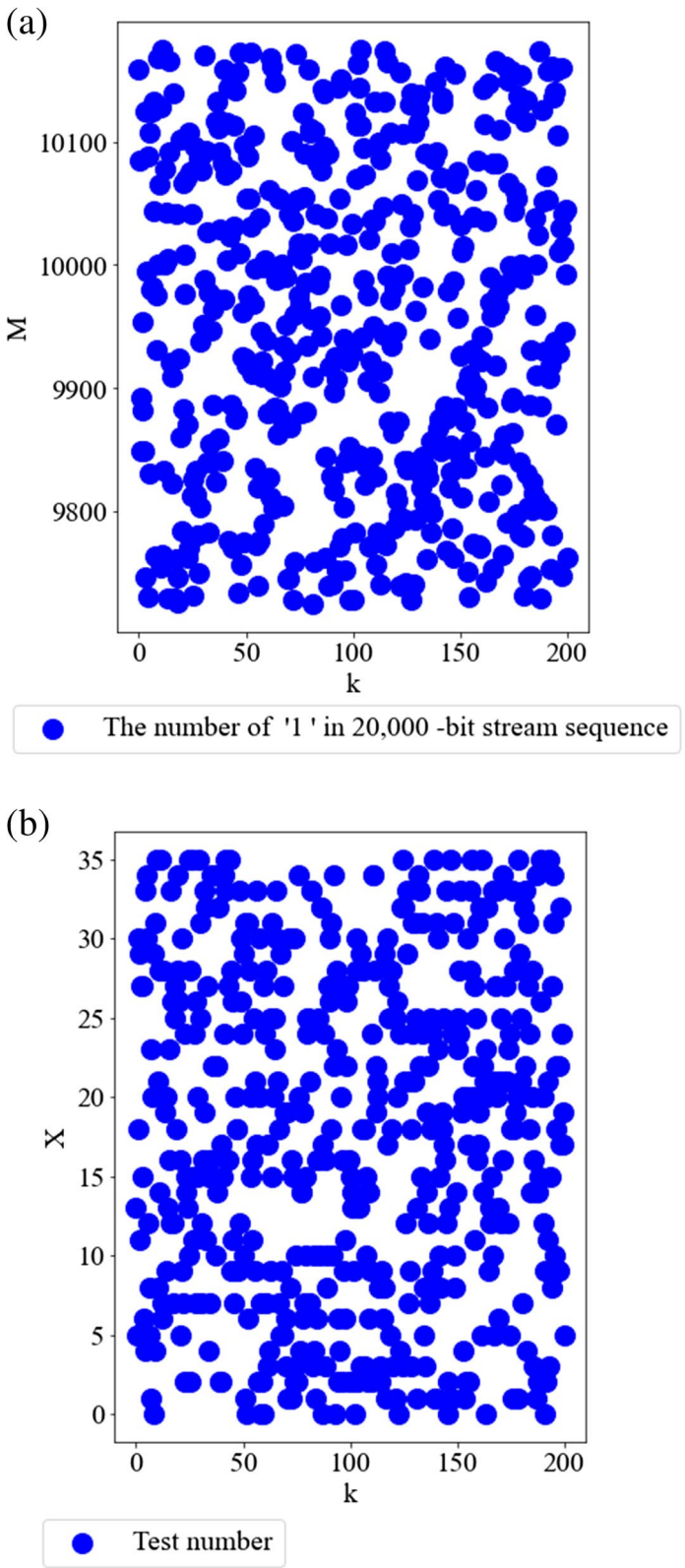


Fig. 3 The result of **a** the Monobit test and **b** the Poker test

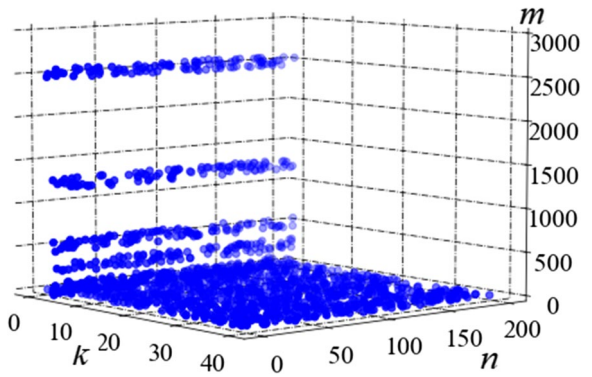


Fig. 4 The result of the Run test and Long Run test

tion entropy reaches the maximum value. Here, the maximum information entropy of the 2048bit sequence is log2048 = 11. It can be seen that the information entropy of the three algorithms is close to 11, approaching the theoretical upper limit. The information entropy of the ZPKG algorithm is greater than that of the LSFR and LC, indicating that the ciphertext generated by it has a higher degree of disorder and a better scrambling effect.

Sensitivity test

(1) Check whether it conforms to the avalanche principle and the mutual correlation characteristics. Randomly change the key by 1 bit, and the obtained 2 KB ciphertext changes by 1.023 KB and the change amount is 49 95%, which shows that the encryption method has a strong avalanche effect. The cross-correlation functions of the obtained two sets of ciphertexts are shown in Fig. 6. A value close to zero indicates that a small change in the key will cause a complete change in the ciphertext.

(2) The generalized avalanche effect is not only the change of the ciphertext caused by the change of the key but also the change of the ciphertext caused by the change of the bits of the plaintext. Extend the experiment in Fig. 6, flip 1 bit of the plaintext, exchange 2 bits of any plaintext, change 3 bits of the key, change 10 bits of the key, and see the bit-flipping ratio of the ciphertext. All experiments were repeated 10 times, and the average value of the results was taken, as shown in Table 3. It can be seen that the ciphertext bit flip ratio is close to 0.5, which meets the strict avalanche criterion (Strict Avalanche Criterion, SAC).

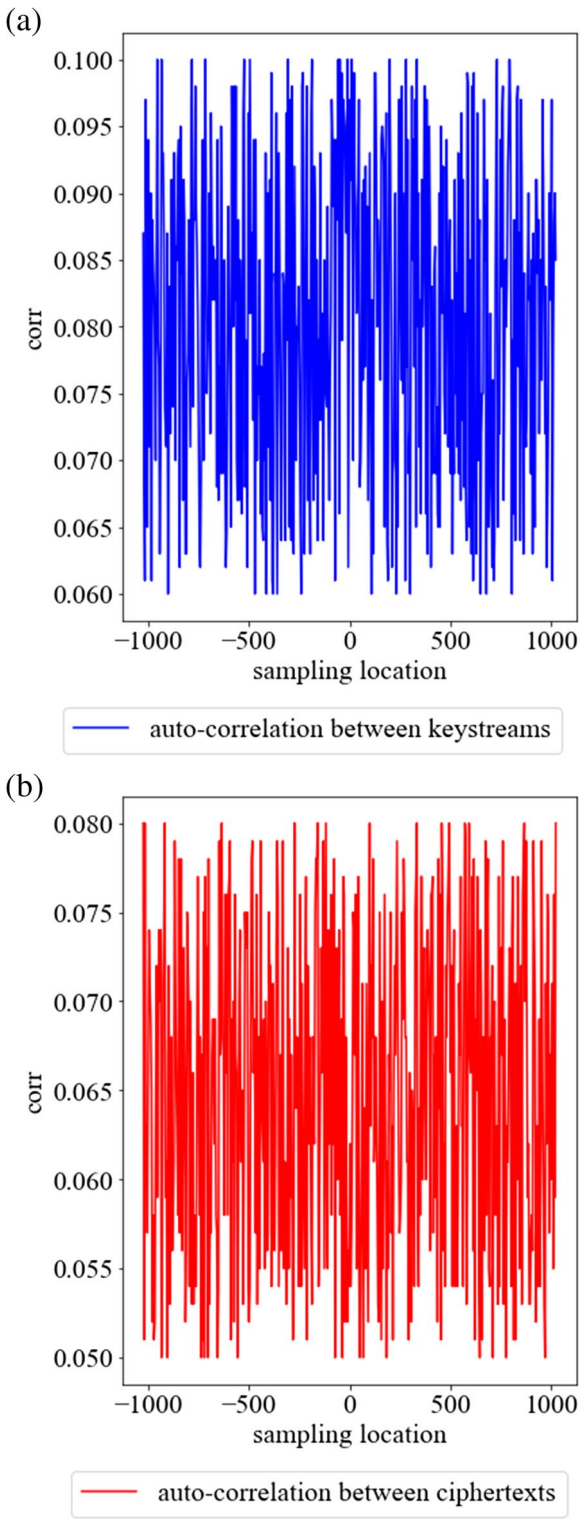


Fig. 5 Autocorrelation of ZPKG

Table 2 Information entropy between different algorithms

Algorithm	Information entropy
ZPKG	10.9992
LSFR	10.9378
LC	10.999

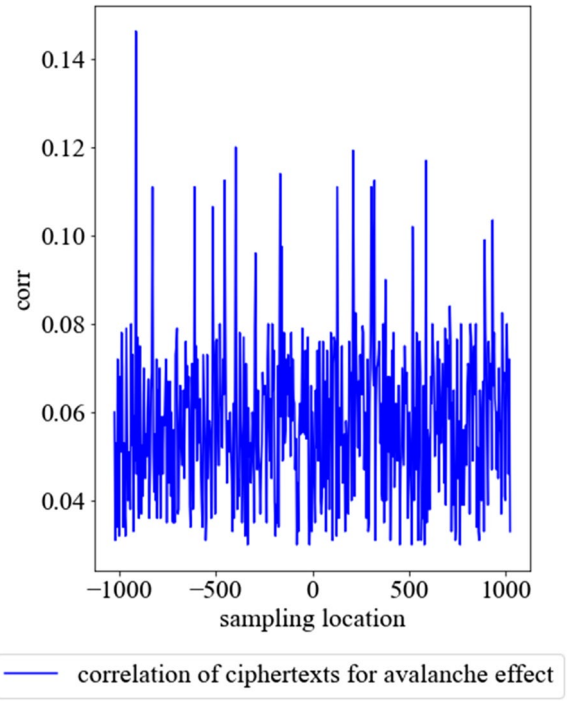


Fig. 6 Test results of the avalanche effect

Table 3 Sensitivity analysis of ZPKG

Operation	Bit flip ratio of ciphertext
Any bit of plaintext flip	0.4953
Swap two bits arbitrarily	0.4951
Key change by 3bit	0.4833
Key change by 10bit	0.4815

Algorithm performance

The simulation environment of this experiment is: Windows Server 2008 R2 Enterprise operating system, the processor is Intel(R) Xeon(R) CPU E7-4860 v4 @ 2.27GHz, RAM 4.0 GB.

In the experimental simulation process, the simulation generates a key stream from 0 to 1 million bits and observes the time consumption of LSFR, LC, and this scheme (ZPKG). The results are shown in Fig. 7.

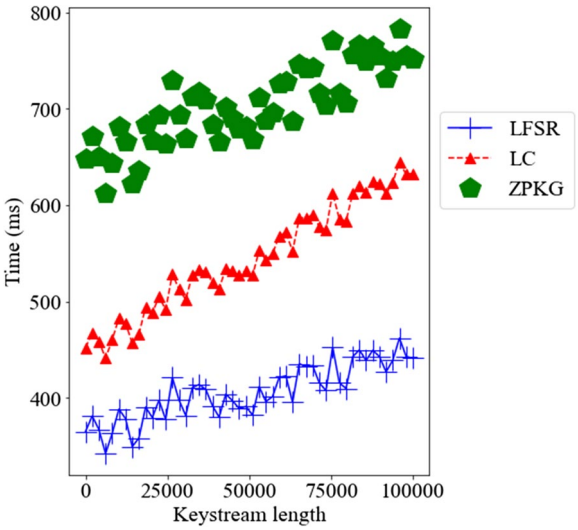


Fig. 7 Time cost for generating the keystream

As the length of the required key stream increases, and the execution time of the algorithm also increases. The time-consuming increase of the LC algorithm with the key stream length is the largest, followed by ZPKG, and LFSR is the smallest. In terms of basic time consuming, the basic time consuming of the ZPKG algorithm is the most, followed by LC, and LFSR is the least. This reflects that the initial preparation time of the ZPKG algorithm is the longest, but as the length of the keystream increases, the time consumption of the ZPKG algorithm increases linearly, and the performance is slightly lower than that of LFSR, but slightly higher than that of LC.

Conclusion

This paper presents a novel stream ciphering algorithm that is suited for cloud-edge communication in IoV. The unpredictability of the keystream is the foundation of stream ciphering. The probability of digit 1 occurring in the center section of the Zeckendorf representation is proven to be constant, which may be utilized to construct a pseudo-random keystream sequence. The pseudo-random sequence created by the linear feedback shift register (LSFR) is periodic and always repeated. The starting value of the sequence generated by chaotic encryption is too sensitive to disturbance, making it easy to be cracked. These two issues are addressed by the proposed algorithm: ZPKG. The keystream generated by ZPKG is not only very random but also endlessly lengthy, with a moderate parameters sensitivity. The experimental results support the above advantages.

Future research directions include:

(1) The ASIC design of this algorithm. In terms of hardware resource utilization and operating speed, this technique is nearly identical to the linear feedback shift register (LFSR). LFSR has the disadvantage of being periodic, therefore it can only be used for the encryption of particular length sequences. The sequences created by ZPKG, on the other hand, are not only random, but also aperiodic, and can yield limitless keystreams. Therefore, ZPKG might be investigated as a replacement for LFSR as a PRBS (pseudo-random binary sequence) generator, which is often used in telecommunication to produce white noise.
(2) Quickly generate the Fibonacci numbers. Fibonacci numbers are required for the Zeckendorf encoding process. The computational complexity ($o(\log n)$) of generating Fibonacci numbers is still high, resulting in slow encoding.

Abbreviations

IoV	Internet of Vehicles
RSU	Road Side Unit
OBU	Short for On Board Unit
DSRC	Dedicated Short-Range Communication
ZPKG	Pseudo-random Keystream Generator with Zeckendorf representation
LFSR	Linear feedback shift register
LC	Logistic Chaos

Acknowledgments
The authors acknowledge Tianqi Cai for language polishing.

Authors' contributions
Yun Wu: data curation, investigation, visualization, and writing. Liangshun Wu: software, validation, methodology, and formal analysis. Hengjin Cai: conceptualization, ideas, funding acquisition, project administration, resources allocation, and supervision. All authors reviewed the manuscript. The author(s) read and approved the final manuscript.

Funding
This work was supported in part by the National Natural Science Foundation of China under Grant 61832014.

Availability of data and materials
The plaintext data used for encryption testing is randomly generated. Relevant experimental results can be requested by email to the corresponding author.

Declarations

Competing interests
The authors declare no competing interests.

Author details
[1]School of Computer Science, Wuhan University, Wuhan 430072, China. [2]School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China.

Received: 28 December 2022 Accepted: 1 March 2023
Published online: 17 March 2023

References

1. Zhou H, Xie H, Zhang H, Zhang H (2021) Parallel remote sensing image encryption algorithm based on chaotic system and DNA coding. J Image Graph 26(05):1081–1094
2. Lekkerkerker CG (1952) Voorstelling van natuurlijke getallen door een som van getallen van Fibonacci. Simon Stevin 29:190–195
3. Zeckendorf E (1972) Représentation des nombres naturels par une somme de nombres de Fibonacci ou de nombres de Lucas. Bull Soc R Sci Liège 41:179–182
4. Kimberling C (1998) Edouard Zeckendorf. Fibonacci Quart 36:416–418
5. Rosen KH (2010) Elementary number theory and its applications. China Machine Press, Beijing
6. Brown J Jr (1964) Zeckendorf's theorem and some applications. Fibonacci Quart 2:162–168
7. Lekkerkerker CR (1951) Voorstelling van natuurlijke getallen door een som van getallen van Fibonacci. In: Stichting Mathematisch Centrum. Zuivere Wiskunde, p 30
8. Yann Bugeaud; On the Zeckendorf representation of smooth numbers, ARXIV-MATH.NT, 2019
9. Idziaszek T (2021) Efficient algorithm for multiplication of numbers in Zeckendorf representation
10. Gessica Alecci; Nadir Murru; Carlo Sanna; Zeckendorf representation of multiplicative inverses modulo a Fibonacci number, ARXIV-MATH.NT, 2022
11. Vukusic I, Ziegler V (2023) On sums of two Fibonacci numbers that are powers of numbers with limited hamming weight. ARXIV-MATH.NT
12. Wu L, Cai H, Liu J, Li Z (2021) Enhancing the anti-cryptanalysis ability and avalanche effect with Zeckendorf representation via FPGA implementation. In: 2021 4th international conference on advanced electronic ...
13. Wu X, Wang D, Kurths J et al (2016) A novel lossless color image encryption scheme using 2D DWT and 6D hyperchaotic system. Inf Sci 349:137–153
14. Kim H, Han J, Cho S (2007) An efficient implementation of RC4 cipher for encrypting multimedia files on mobile devices. In: Proceedings of the ACM symposium on applied computing. ACM, Seoul, pp 1171–1175
15. Berger TP, Minier M, Pousse B (2009) Software oriented stream ciphers based upon FCSRs in diversified mode, international conference on cryptology in India. Springer, New Delhi, pp 119–135
16. Hosseini S (2018) Fingerprint vulnerability: a survey. In: 2018 4th international conference on web research (ICWR)
17. Zhou L, Tan F (2019) A chaotic secure communication scheme based on synchronization of double-layered and multiple complex networks. Nonlinear Dyn 96:869–883
18. Djamaluddin B, Ferianto T, Akbar H (2020) End to end data security challenges in real-time drilling data environment - from data transfer to analytics
19. Chen W, Chen G, Zhao Y, Zhang J (2021) Security vulnerability and encryption technology of computer information technology data under big data environment. J Phys Conf Ser 1800(1):012012
20. Lin Z (2013) The transformation from the Galois NLFSR to the Fibonacci configuration. In: International conference on emerging intelligent data and web technologies. IEEE, Xi'an, pp 335–339
21. Mansouri SS, Dubrova E (2013) An improved hardware implementation of the quark hash function. International workshop on radio frequency identification: security and privacy issues. Springer, Graz, pp 113–127
22. RC4 Encryption Algorithm. Available: https://en.wikipedia.org/wiki/RC4. Accessed 3 Jan 2023
23. Mantin SA (2001) Weaknesses in the key scheduling algorithm of RC4, Lecture Notes in Computer Science. In: Revised papers from the 8th annual international workshop on selected areas in cryptography, vol 2259, pp 1–24
24. Wu C, Kuo CJ (2005) Design of integrated multimedia compression and encryption systems. IEEE Trans Multimed 7(5):828–839
25. Sreelaja NK, Pai GAV (2012) Stream cipher for binary image encryption using ant colony optimization based key generation. Appl Soft Comput 12:2879–2895

26. Golomb SW (1967) Shift register sequences: USA[P]. Holden-Day Inc., San Francisco
27. Filipponi P, Wolfowicz W (1987) A statistical property of nonadjacent ones binary sequences. Note Recensioni Notizie 36(3):103–106
28. Mansour T (2002) Combinatorial identities and inverse binomial coefficients. Adv Appl Math 28:196–202
29. National Institute of Standards and Technology (2001) FIPS 140–2-2001 security requirements for cryptographic modules. American National Standards Institute, Washington DC

Publisher's Note