electronics

MDPI

Article

KryptosChain—A Blockchain-Inspired, AI-Combined, DNA-Encrypted Secure Information Exchange Scheme

Pratyusa Mukherjee [1], Chittaranjan Pradhan [1,*], Hrudaya Kumar Tripathy [1] and Tarek Gaber [2,3]

[1] School of Computer Engineering, Kalinga Institute of Industrial Technology (KIIT) Deemed to Be University, Bhubaneshwar 751024, India
[2] School of Science, Engineering & Environment, University of Salford, Salford M5 4WT, UK
[3] Faculty of Computers and Informatics, Suez Canal University, El Salam District, El Sheikh Zayed 8366004, Egypt
* Correspondence: chitaprakash@gmail.com

Abstract: Today's digital world necessitates the adoption of encryption techniques to ensure secure peer-to-peer communication. The sole purpose of this paper is to conglomerate the fundamentals of Blockchain, AI (Artificial Intelligence) and DNA (Deoxyribonucleic Acid) encryption into one proposed scheme, KryptosChain, which is capable of providing a secure information exchange between a sender and his intended receiver. The scheme firstly suggests a DNA-based Huffman coding scheme, which alternatively allocates purines—Adenine (A) and Guanine (G), and pyrimidines—Thymine (T) and Cytosine (C) values, while following the complementary rule to higher and lower branches of the resultant Huffman tree. Inculcation of DNA concepts makes the Huffman coding scheme eight times stronger than the traditional counterpart based on binary—0 and 1 values. After the ciphertext is obtained, the proposed methodology next provides a Blockchain-inspired message exchange scheme that achieves all the principles of security and proves to be immune to common cryptographic attacks even without the deployment of any smart contract, or possessing any cryptocurrency or arriving at any consensus. Lastly, different classifiers were engaged to check the intrusion detection capability of KryptosChain on the NSL-KDD dataset and AI fundamentals. The detailed analysis of the proposed KryptosChain validates its capacity to fulfill its security goals and stands immune to cryptographic attacks. The intrusion possibility curbing concludes that the J84 classifier provides the highest accuracy of 95.84% among several others as discussed in the paper.

Keywords: information exchange; DNA encryption; blockchain technology; secure communication; intrusion detection system

Citation: Mukherjee, P.; Pradhan, C.; Tripathy, H.K.; Gaber, T. KryptosChain—A Blockchain-Inspired, AI-Combined, DNA-Encrypted Secure Information Exchange Scheme. *Electronics* **2023**, *12*, 493. https://doi.org/10.3390/electronics12030493

Academic Editor: Flavio Canavero

Received: 29 December 2022
Revised: 10 January 2023
Accepted: 14 January 2023
Published: 17 January 2023

1. Introduction

Peer-to-peer communication has become a part of our daily life. Suppose a sender wishes to send a message to his friend, then, he will definitely want the secrecy of the message to be restricted only between them and not be revealed to any unauthorized entity; the content of the message must be as it is and the message should be exchanged within the stipulated interval of time. The most common solution is to resort to encoding, which converts the original message into a codeword using a cryptographic key that is next transmitted to the receiver. Only the sender and the receiver are aware of this key and, therefore, the codeword seems incomprehensible to any intruders. Encryption thus prohibits adversarial attacks on information during transmission and storage by safeguarding its confidentiality, integrity and availability. Traditional encryption schemes mostly rely on binary values; thus, their exponential power is 2. DNA encryption [1] is the new innovative technique to perform encryption of DNA sequences comprising the four nitrogenous bases—A, T, C and G. This produces the exponential power of 4. Thus, DNA encryption methodologies are eight times stronger than the traditional schemes. A non-vulnerable

transmission of data is guaranteed by the combination of the chemical characteristics of biological DNA sequences and classical cryptography [2,3].

Blockchain [4,5] technology is another propitious field that is finding wide usage in the mainstream areas of security, trust and privacy [6–10]. It is fundamentally a distributed database based on a chain data structure that links blocks using the concept of hashing [11,12]. Each successive block stores the hash of its previous block. As a result, any sort of tampering or counterfeiting is immediately noticed. Also, information once uploaded to the blockchain is immutable, which again prevents any sort of repudiation. The decentralization feature [13] and consensus mechanism [14] of the Blockchain ensures trust, security, transparency and the traceability of information shared across any network. However, the major drawback of real Blockchains includes complex and expensive implementation. Also, blockchain and smart contracts go hand in hand, which are difficult to achieve, update, modify and are also time-consuming and possess scalability issues.

Artificial intelligence (AI) [15] techniques can further improve overall security performance and provide better protection from an increasing number of sophisticated threats. Artificial intelligence has avid applications in the field of security and blockchain such as energy optimization, collaborative learning, intrusion detection, authentication validation, hash calculation, quick mining, secure gate-keeping etc. [16]. AI models can provide chaos, randomness, and many other properties, all of which are required by cryptosystems. This paradigm is termed AI-influenced cryptography (AIIC) [17]. On the other hand, AI can also be evolved by inculcating the concepts of cryptography into it, which is termed crypto-influenced AI (CIAI) [18]. Cryptography hugely relies on the confusion and diffusion of the relationship between the plaintext, ciphertext and key. Ideally, the key and ciphertext should be entirely devoid of any type of pattern. One of the principal features of AI is its ability to recognize patterns within complex data, which in turn benefits information security. Another major application of AI is in intrusion detection as AI-based IDS systems are superior in their ability to autonomously identify threats.

This paper, therefore, tries to amalgamate the benefits of DNA encryption, Blockchain technology and artificial intelligence into the proposed KryptosChain scheme. The sole contribution of this paper is to propose a secure information exchange scheme between two parties, which is subdivided into two broad steps. First, the original input is converted in the form of a DNA string using our proposed DNA-based Huffman Coding scheme. The basic feature of a Huffman code [19], assigning variable length codes and allotting shorter codes to more frequent characters, is exploited here as well. The refinement that this proposal suggests is what value to assign to the higher and lower branch of the Huffman tree instead of the traditional assigning of a 1 and 0, respectively. After successful generation of the cipher form, to share it with the validated receiver, a blockchain-inspired protocol has been suggested, which poses a refinement on the well-acclaimed Diffie-Hellman Key exchange protocol [20]. Each of the successive blocks of the proposed protocol contains the hash of its previous block. This enhances the security as even a slight change in the block modifies its hash, which will be reflected in its successive blocks, and hence any kind of contamination is easily noted. Thusly, the proposed scheme eliminates the susceptibility of man-in-the-middle attacks in the information exchange. The authentication of the two parties involved in the communication and any sort of intrusion is condemned by the application of AI.

Thus, the prime highlights of this paper are:

- A proposed DNA-based Huffman Coding Scheme. It considers the real-time occurrence of every distinct symbol in the plaintext to determine their frequency distribution. In contrast to assigning a 1 to the higher branch and 0 to the lower branch after adding the two least frequent symbols, the proposed scheme alternatively assigns a purine and pyrimidine value to the high and low branch. A different Huffman tree needs to be derived each time to get the corresponding codes, thusly enhancing the security. The variable length of the codes also makes them less guessable and immune to attacks.

- A Blockchain-inspired refinement on the Diffie-Hellman Key exchange protocol is proffered to transfer the cipher information to the intended receiver. Blockchain technology, due to its highly secure and decentralized nature, is predominately used for secure transmission and storage. However, they necessitate possession of cryptocurrencies, writing smart contracts and deploying them to facilitate their many possible functions. Therefore, this paper puts forward a blockchain-inspired scheme that transmits fixed-sized blocks of the original message to the genuine receiver. The trusted third party is only involved to authenticate the sender and receiver, and unknowingly assist them to establish the shared secret key. He only knows the hash of the public key and cannot obtain the actual public key as hashes are one-way The actual message exchange is also safeguarded from the trusted third party as they are encrypted by the intended receiver's public key, which can only be decrypted with a corresponding private key. Thus, involved parties can exchange information securely via the proposed scheme. An AI-influenced intrusion detection system to further ensure secure communication between the sender and intended receiver. Different classifiers were used to train and test the proposed IDS system such as NB (naive Bayes), logistic, MLP (multi-layer perceptron), SMO (sequential minimal optimization), IBK (instance-based), and J48 on the NSL-KDD dataset.
- The paper first provides a brief introduction to DNA encryption followed by Blockchain technology and artificial intelligence. It also discusses the possibility of coalescing all these three technologies into the proposed KryptosChain scheme. Section 2 focuses on related work of existing research. The proposed methodology is illustrated in Section 3, which first shows the basic block diagram of the overall steps involved and their descriptions. All the results obtained have been categorically demonstrated in Section 4. Section 5 provides the analysis of schemes proffered by this paper. The conclusions drawn and future scope of work are presented in Section 6.

2. Related Work

The existing literature research supporting the aim of this paper was conducted strategically and systematically followed by detailed analysis as represented below.

2.1. Existing DNA-Based Encryption Scheme

The idea of adding fundamentals of DNA into encryption has been recognized as a feasible technology with a new goal of improving algorithm robustness. The related work can further be subdivided into three categories depending upon the types of algorithms used for the encryption—substitution-based DNA encryption schemes, biological operations-based DNA encryption schemes, mathematical–biological operations-based DNA encryption schemes.

Substitution-based methods [21–23] utilize a pre-decided look-up table or DNA dictionary to perform the encipherment process. An example look-up table is shown in Table 1 for letters (case insensitive), digits, and some symbols. These methods are the most non-complicated and simple DNA encryption techniques.

Table 1. An example DNA Encryption Look-up Table.

A = CTG	B = ACC	C = GAC	D = GAT	E = GCG	F = AGT	G = ATG	H = CGT
I = AAG	J = AGC	K = AGG	L = TGC	M = TCA	”” = GCT	O = GAA	P = GTC
Q = ACA	R = CAC	S = AGT	T = TTA	U = ATA	V = CTT	W = CCA	X = CTA
Y = AAA	Z = CTT	0 = ACT	1 = AGC	2 = TAG	3 = GCA	4 = GAC	5 = AGA
6 = TTC	7 = ATT	8 = AGC	9 = GCT	, = CCC	. = GTC	! = GCT	? = CCT

Biology-involved algorithms [24–26] use only rigorous biological operations to perform the encryption procedure, such as polymerase chain reaction, transcription, translation, microdotting, DNA fragmentation, DNA hybridization etc. They require minimal human intervention and therefore are comparatively more secure. Initially, using a sim-

ple substitution technique, the DNA encoded form is obtained. Biological operations are then applied on them to get the final cipher DNA sequence form. Zhang et al. [27] proposed a new DNA cryptography algorithm based on the bio puzzle and DNA chip technology. The DNA chips and keys are finally sent to the receiver. Wang et al. [28] proposed a technique to hide messages in living organisms using DNA encoding, as well as DNA recombinant technology.

The third category of algorithms comprises those that utilize both mathematical as well as biological operations to perform the encryption [29–31]. Mathematical operations involve the usage of symmetric or asymmetric cryptographic keys and the biological operations provide an additional layer to them, thus making them the most secure DNA-based technique. Some algorithms simply DNA encode the intermediate ciphertext on the basis of substitution techniques.

2.2. Analysis of Existing DNA-Based Encryption Scheme

Table 2 offers the performance evaluation of the three categories of DNA-dependent encryption methods in terms of cryptographic key involvement and type, and their encryption time and major limitations.

Table 2. Performance evaluation of existing DNA-based Encryption schemes.

Category	Involvement of Key	Encryption Time	Limitations
Substitution-based Schemes	No	Minimum	• Prone to statistical attacks—Known plaintext attack, chosen ciphertext attack. • Size of ciphertext is huge as compared to size of plaintext. • Same plaintext will be encrypted to same ciphertext each time because of the usage of same substitution process.
Biological Operations-based Schemes	Biological information is used as key-position of exon-intron, primer values etc.	Maximum	• They are computationally strenuous and time-consuming. • Biological operations are extremely critical to implement and are economically cumbersome. • They sometimes work beyond the control of the sender and receiver. • The correct plaintext may not be decrypted even after applying the same steps as in the encryption process.
Mathematical–Biological Operations-based Schemes.	Keys are generated by stringent mathematical calculations	Moderate	• Complex and rigorous calculations are involved

2.3. Existing Blockchain-Based Information Exchange Schemes

Partala [32] proposed a combination of steganography and a Blockchain-based secure communication scheme over covert channels. The study formulated the notion of payments in which the hidden message is camouflaged and indistinguishable from random payments.

Guziur et al. [33] developed a Blockchain communication protocol in addition to the SHA-3 based seed values between a client application and server. Their entire proposal is segregated into secret seed generation, followed by its transmission from the client side, partitioning the data to be transferred and creating the Blockchain. Next, they have performed the actual data transfer, which is then verified and portions of data are merged back to get the original piece of data.

Saritekin et al. [34] proffered Cryptouch, a prototype communication application relying on fundamentals of Blockchain and the interplanetary file systems (IPFS) between

two users on the same network. Menegay et al. [35] produced a scheme in which the email server is added into an existing Blockchain to enable secure email exchange between the involved parties.

Naz et al. [36] suggested another data sharing platform utilizing similar concepts. They proposed extensive data sharing, retrieval and reviewing steps to ensure secure sharing of vital data, as well handle disputes if any. Their scheme involves actual monetary deposits.

Bi et al. [37] suggested an accelerated methodology of message transmission in an existing Blockchain network by every time choosing a closest neighbor and then propagating it in the entire network. It thus leads to minimal latency and is highly energy efficient. Ellewala et al. [38] developed a scheme to deploy a private Blockchain that is restricted to a particular enterprise and crucial information is updated into this private Blockchain so that only the associated members of the enterprise can access it. To enhance the security, an encrypted version of the information should be added in the Blockchain

Singh et al. [39] proposed a Blockchain instant messaging application. In this, a user first generates the public/private key pair during installation. Next, the mobile network operator (MNO) issues a digital certificate and stores it on a public Blockchain. Any user can fetch the certificate for its receiver from the server and enjoy secure communication using a ratchet forward encryption mechanism.

Khacef and Pujolle [40] proposed a Blockchain-based messaging model using smart contracts to verify the identities of the parties involved and their associated public keys. As per their scheme, each user has to undergo a registration procedure followed by a smart contract-based verification. Only after authentication has been ensured, the sending and receiving of the messages can be performed. Although this model eliminates a central authority, deployment of smart contracts hampers the performance of the system. Designing of the appropriate smart contract plays a crucial role as it is not possible to immute an existing contract under any circumstances.

2.4. Analysis of Existing Blockchain-Based Information Exchange Schemes

In Table 3, we study the above-mentioned schemes to identify their merits and demerits.

Table 3. Performance evaluation of existing Blockchain-based Information Exchange Schemes.

Paper Title	Merits	Demerits
Provably secure covert communication on blockchain—Partala [32]	• Uses a steganography-based public blockchain • Original payments are camouflaged in random payments	• Scalability issues • Real cryptocurrency involved
Light blockchain communication protocol for secure data transfer integrity—Guziur et al. [33]	• Uses an SHA3-based Public Blockchain • More secure	• Time-consuming due to multiple steps • Real cryptocurrency involved
Blockchain-based secure communication application proposal—Sarıtekin et al. [34]	• Uses an IPFS-based Blockchain • More secure	• Scalability issues. • Real cryptocurrency involved
Secure communications using blockchain technology—Menegay et al. [35]	• Uses a Public Blockchain • Email server added in an existing Blockchain	• Difficult to deploy • Real cryptocurrency involved
A secure data sharing platform using the blockchain and interplanetary file system—Naz et al. [36]	• Uses a Public Blockchain • Digital assests are shared and delivered	• Economically cumbersome as asks for money to upload and retrieve data • Real cryptocurrency involved

Table 3. *Cont.*

Paper Title	Merits	Demerits
An accelerated method for message propagation in blockchain networks—Bi et al. [37]	• Uses a Public Blockchain • Fast propagation of message via closest neighbor	• Secrecy and privacy hampered as a nearest neighbor also gets a copy of the data even if he is not the intended receiver • Real cryptocurrency involved
Secure Messaging Platform Based on Blockchain—Ellewala et al. [38]	• Uses Private Blockchain • Restricted to a single enterprise and thus is secure	• Expensive. • Real cryptocurrency involved
Blockchain-enabled end-to-end encryption for instant messaging applications—Singh et al. [39]	• Uses MNO involving Private or Public Blockchain • Each user uploads his public key certificate into the Blockchain, thus enhancing the security	• Highly dependent on MNO • Real cryptocurrency involved
Secure peer-to-peer communication based on Blockchain—Khacef and Pujolle [40]	• Uses Public Blockchain • Removes dependency on certification authority, and thus is more secure	• Scalability issues. • Real cryptocurrency involved

2.5. Existing AI-Based Cryptographic Schemes

The most predominant threat on the internet is that of distributed denial of service (DDoS). Existing research [41,42] recommends that AI can help to effectively classify DDoS attack traffic and normal traffic using random forest tree and I Bayes to finally detect it. An AIMM (artificial intelligence merged methods) framework was proposed by Jaszcz and Poap [43]. The three modules that make up the solution are analyzing the incoming data, categorization and decision-making. The decision-making module gathers the probabilities through AI techniques such as neural networks and k-nearest neighbors by finding the weighted aggregates.

IDS (intrusion detection system) based on AI [44–46] makes an effort to understand the typical patterns of network traffic and spot abnormalities and deviations based on algorithmic departures from those known patterns. IDS continuously monitors the actions of a system to identify possibility of an attack. Once it detects any probable attack, it generates alarms to signal necessary steps must be rendered to mitigate its consequences. The input to the IDS can be traffic statistics, information gathered from packet headers and its content, information from hosts like their process behavior, system call logs, application logs, file system modifications etc. The output of the IDS could be a binary label–normal or attack, or multi-valued indicating different types of attacks for each input or a series of inputs. From a machine learning perspective, this problem can be formulated as a classification problem. Thus, it needs a labeled dataset of normal and attack inputs for training. After a model is trained, it can be deployed to take decisions on new data from the system.

Malware propagation by adversaries is another major concern and to curb its proliferation, artificial intelligence is again having a wide outreach [47,48].

Marwala and Xing [49] studied the alliance of AI and blockchain and offered a brief overview about how artificial intelligence could be used to deliver bug-free smart contracts [50].

A major limiting factor on safe integration of AI in the real world cryptographic utilities is the quality of the data used to train these systems. Malicious data cause AI systems to generate incorrect outputs. Scalability is another crucial issue. At all costs, AI with human intervention is the most appropriate solution.

3. Proposed Methodology

The suggested KryptosChain has three broad steps. First is the cipher information generation followed by its secure transmission to the intended receiver. An AI-based methodology is also proposed to detect any kind of intrusion. In the last subsection, the corresponding decryption scheme is also put forward.

3.1. Proposed Cipher Information Generation Scheme

This section represents the nitty-gritty of encoding the original information in the form of a DNA string, which will ultimately act as the intermediate ciphertext of the proposed overall information exchange scheme. It provides an elaborate description of the steps of our proposed DNA-based Huffman coding scheme. Huffman coding saves a lot of storage space as it generates variable length codes. It also assigns shorter codes for symbols that appear more frequently in the input string. However, traditional Huffman codes mainly involve binary values only. Thus, they demand refinement to suit the DNA cryptographic schemes.

The flowchart and algorithm of the DNA-based Huffman coding is depicted in Figure 1 and Algorithm 1. The first step of the proposed scheme is to find the frequency of distinct symbols in the original plaintext. Next, they are arranged in increasing order of frequency and if the same frequency then in alphabetical order. Each time, the least two frequencies are added to form a subgroup. The sum of their frequency is the root. A purine value(i.e., A or G) is allotted to the higher branch instead of a "1". The corresponding complementary pyrimidine value (i.e., T or C) is allotted to the lower branch instead of a "0". This process continues alternatively until all symbols are merged. Finally, all the nucleotides are read from the root to that symbol to generate its corresponding DNA-encoded string.

Algorithm 1: Proposed DNA-based Huffman Coding Scheme

Generating the Huffman Tree:

1. Create and initialize a Priority Queue consisting of each unique symbol in the original message.
2. Sort in ascending order of their frequencies.
3. For all the unique symbols:
4. create a new_node
5. get minimum_value from Queue and set it to higher child of new_node
6. get minimum_value from Queue and set it to lower child of new_node
7. calculate the sum of these two minimum values as sum_of_two_minimum
8. assign sum_of_two_minimum to the value of the new_node
9. insert new_node into the tree
10. return root_node

Assigning Values

1. For all the nodes alternatively:
2. assign A or G to the higher child of the new_node and T or C to the lower child of the new_node

Reading the codewords

1. Start from the root_node and traverse towards a unique symbol
2. Note the assigned value from right to left

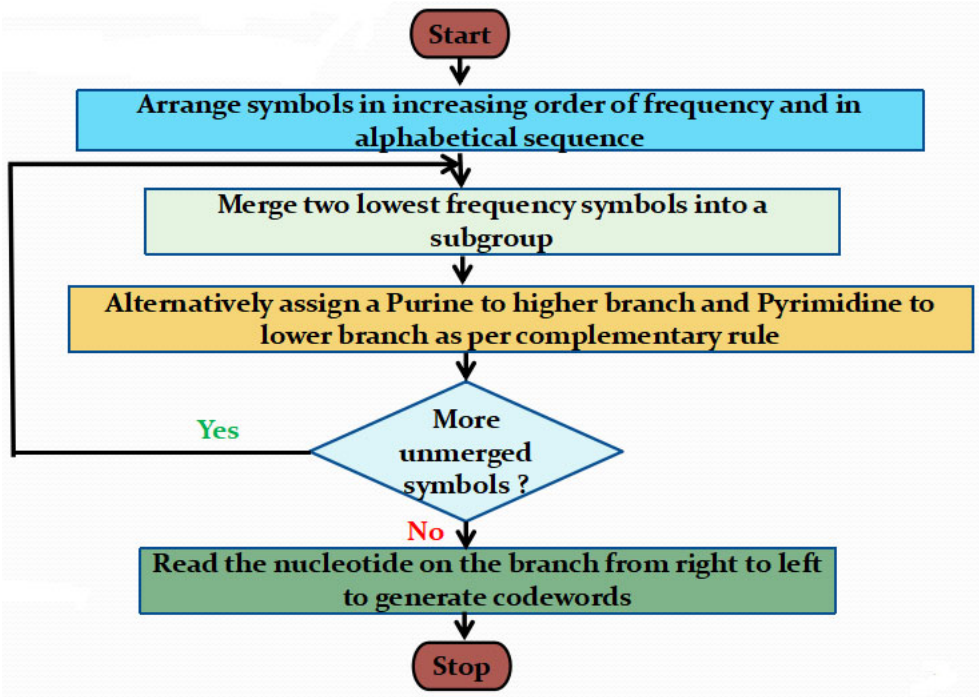

Figure 1. Flowchart of Proposed DNA-based Huffman Coding Scheme.

3.2. Proposed Cipher Information Transmission Scheme

A blockchain-inspired protocol that represents an improvement on the well-known Diffie-Hellman Key exchange protocol has been proposed after the cypher form was successfully generated and ready to be shared with the validated receiver. The hash of the current block is calculated and stored in its succeeding block. This improves security because any form of contamination is quickly detected because even a small modification in the block will change its hash, which will be mirrored in its next blocks. Thus, the suggested approach renders the information exchange immune to man-in-the-middle attacks. There are six phases in the total plan. Kyrios, which is the Greek word for Lord, is a reliable third party who is solely used to help with the sender and receiver's successful registration and authentication. The entire information is broken into blocks of fixed size and each block of KryptosChain by default stores the hash of its previous block as inspired from the Blockchain.

The following variables have been used in this section:
K_{Pb}: Public Key
K_{Pr}: Private key
h(x): Hash of x
p: Large prime number
g: Large prime number and $g \neq p$
x: Sender's random secret number
y: Receiver's random secret number
K_S: Sender's Key
K_R: Receiver's Key
T: Timestamp
M(x): Metadata of x
K_1: Key calculated by sender on basis of K_R
K_2: Key calculated by receiver on basis of K_S
K: Shared secret key established between the sender and receiver

3.2.1. Phase 1: Registration Process

Each user, whether a transmitter or a receiver, must correctly register. They will use an asymmetric approach to generate their public-private (KPb-KPr) key pair. The user will then communicate to Kyrios the hash of the public key h (KPb) while keeping the private key a secret. The SHA-256 algorithm is used to generate the hash values. After successful registration, Kyrios will create a special User ID and provide it back to the user. Additionally, Kyrios will save the results in a look-up table for later use. This phase is depicted diagrammatically in Figure 2.

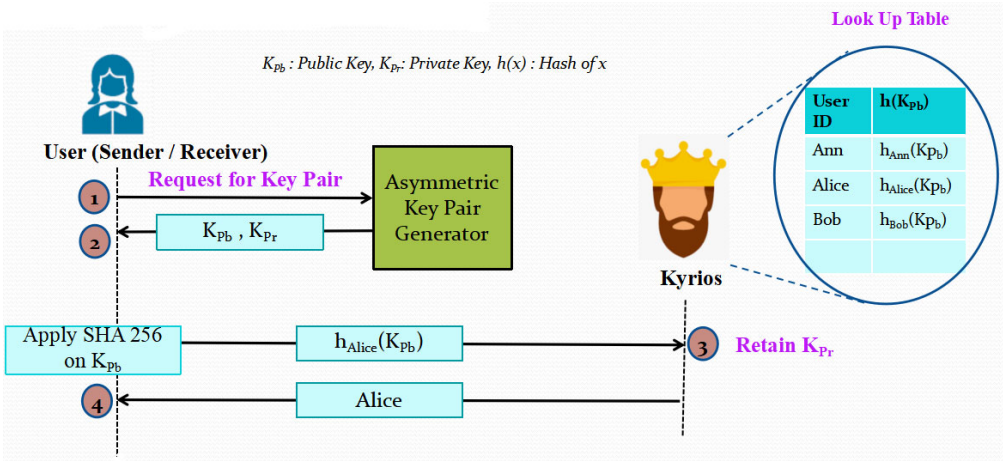

Figure 2. Illustration of Phase 1: Registration Process.

3.2.2. Phase 2: Sender Authentication

Sender must log into KryptosChain using a special user ID and hash of his public key to transmit a message. The hash Kyrios just received will be compared to the hash in his look-up table. Only in the event that the hashes match will the sender be given access. As a result, Kyrios has verified the sender's identity, allowing them to access KryptosChain to add or read blocks. The block diagram of the sender authentication process is shown in Figure 3.

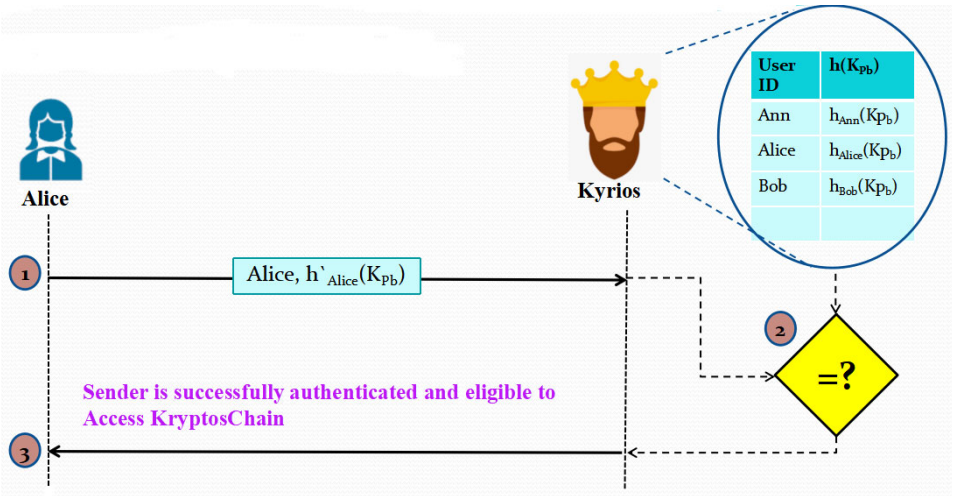

Figure 3. Illustration of Phase 2: Sender Authentication.

3.2.3. Phase 3: Genesis Block Generation Process

In this phase, the first block of the KryptosChain will be created. Alice chooses two large prime numbers p and g and also a secret random number x. She then calculates the sender's key KS = g^x mod p. After this, she sends the user ID of her intended receiver, p,

g and KS values to Kyrios. Kyrios will immediately Timestamp the contents it receives from Alice and return her the hash of the public key of the receiver. Next, Alice will lock the collection of the Timestamped contents she just received and the metadata of the message she is trying to send the receiver with the receiver's public key hash to Kyrios. It is the responsibility of Kyrios to upload this encrypted block as the first or genesis block of KryptosChain. The hash of this block is also immediately calculated and as per the fundamentals of a blockchain, the previous hash value for the genesis block will be null. As per the user ID mentioned by Alice, Kyrios will also inform her desired Receiver that some Alice is trying to contact him. The diagrammatic representation of this step is illustrated in Figure 4.

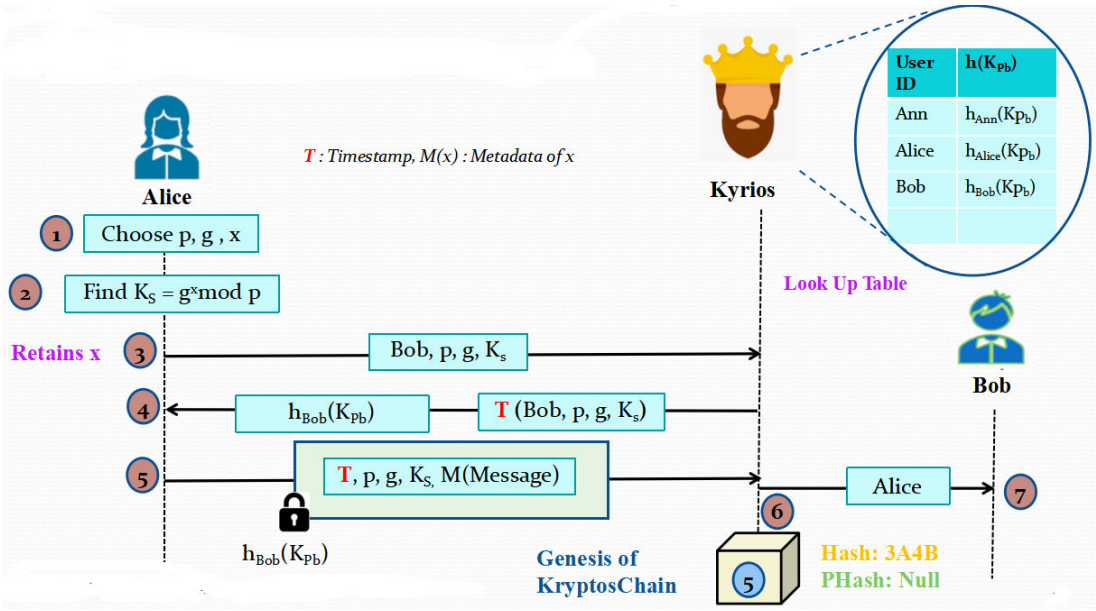

Figure 4. Illustration of Phase 3: Genesis Block Generation Process.

3.2.4. Phase 4: Receiver Authentication

Figure 5 illustrates how Kyrios will authenticate the receiver in a manner similar to how it authenticates the sender. The hash of the sender's public key is supplied to him following his validation.

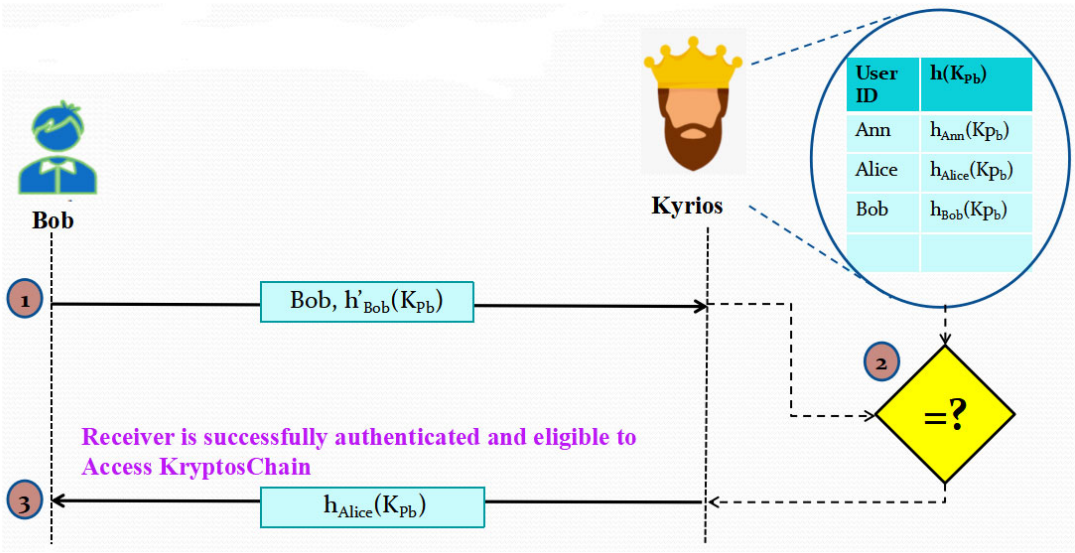

Figure 5. Illustration of Phase 4: Receiver Authentication Process.

3.2.5. Phase 5: Second Block Generation Process

As per Figure 6, the next phase begins with Bob either expressing his willingness or reluctance to continue communication with Alice to Kyrios. If he is reluctant, then no more further steps need to be performed. If he wishes to continue interacting with Alice, he will access KryptosChain and access the contents currently present there by first decrypting it using the hash of his corresponding private key. After this, he will choose a secret random number y and retain it with him. Then, he will calculate the receiver's key $K_R = g^y \bmod p$. His next task is to send this K_R value and the metadata of his response to Kyrios. Kyrios will Timestamp this content and add this block into KryptosChain.

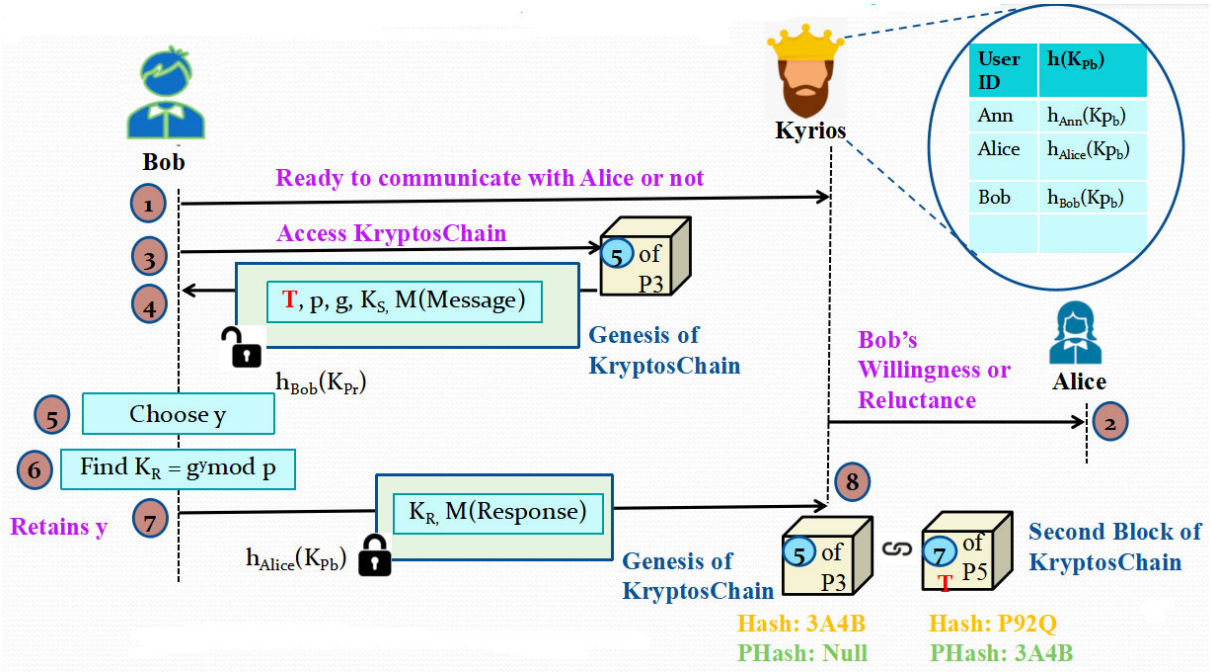

Figure 6. Illustration of Phase 5: Second Block Generation Process.

3.2.6. Phase 6: Actual Information Exchange Process

At this stage, since both Alice (Sender/Transmitter) and Bob (Receiver) are already validated, the responsibility of Kyrios ends. Henceforth, both of them will continue communication without any intervention of Kyrios. Alice will extract the contents of the second block and decrypt it using the hash of his private key. He will calculate a value K_1 using the value of K_R he just found as per Equation (1). Simultaneously, Bob will also find a value K_2 using the value of K_S he has received in Phase 5. This calculation is illustrated in Equation (2).

$$K_1 = K_R{}^x \bmod p = (g^y)^x \bmod p = g^{yx} \bmod p \quad (1)$$

$$K_2 = K_S{}^y \bmod p = (g^x)^y \bmod p = g^{xy} \bmod p \quad (2)$$

Using Equations (1) and (2), we see that mathematically $K_1 = K_2 = K$. This K is nothing but the shared secret key for all further exchange of blocks between the sender and the receiver. Thus, this model proposes a refinement on the famous Diffie-Hellman key exchange protocol by inculcating the concepts of Blockchain into it. After the shared secret key has been established with the aid of Kyrios between the sender and receiver, the actual communication between them happens through KryptosChain. The original message is broken down in fixed sizes of blocks pre-decided by the parties involved, and they are added into the Kryptoschain. To enhance the security, everything is encrypted using the recently established shared secret key K as demonstrated in Figure 7. At this stage, Kyrios is no longer involved in the exchange so the messages are also free from any third party involvement.

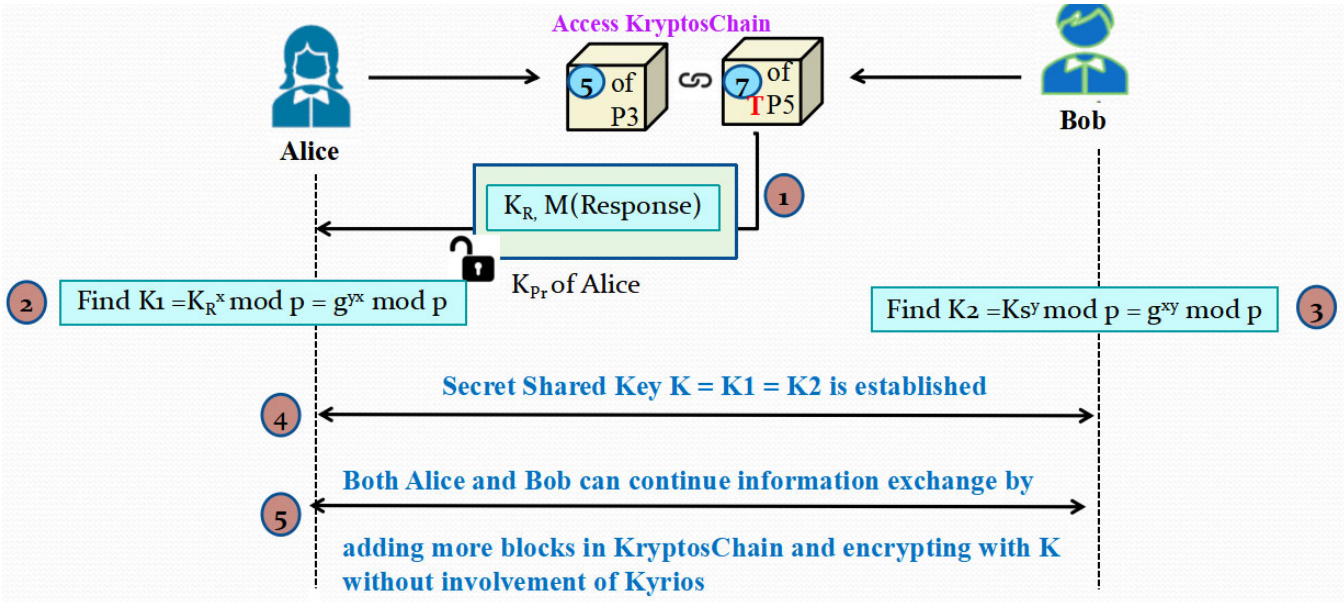

Figure 7. Illustration of Phase 6: Actual Information Exchange Process.

3.3. Proposed Intrusion Detection Scheme

In order to further ensure secure communication between the sender and intended receiver, an IDS guards the overall model. The basic working flowchart of the proposed Intrusion Detection System is shown in Figure 8.

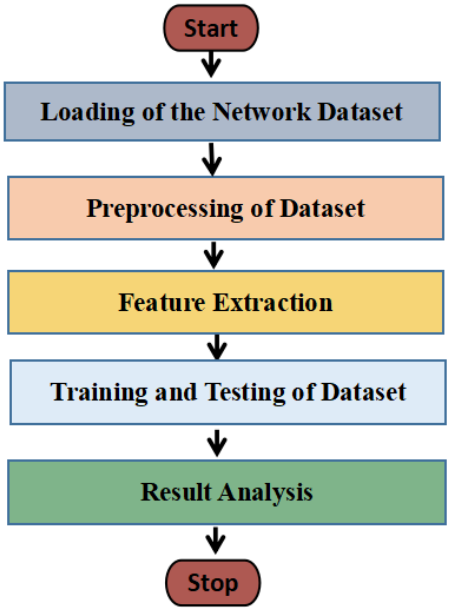

Figure 8. Flowchart of Intrusion Detection Scheme.

The first task it to choose an appropriate dataset. One of the earliest public datasets for IDS is KDD 99 [51] with 43 features and attacks of four categories. The NSL-KDD [52] dataset overcomes the deficiencies in the KDD 99 set by removing duplicate records and selecting records that are difficult to classify. This proposal thus uses the NSL-KDD dataset. This data set is comprised of four subdata sets: KDDTest+, KDDTest-21, KDDTrain+, KDDTrain+_20 Percent. The data set contains 43 features per record, where 41 refers to the traffic input itself and the last two are labels—whether it is a normal or attack and Score, which gives the severity of the traffic input itself. Within the chosen dataset exists four different categories of attacks: denial of service (DoS), probe, user to root (U2R), and re-

mote to local (R2L). The goal of a DoS attack is to stop traffic going to and from the target system. Because of the unusually high volume of traffic, the IDS shuts itself off in defense. This stops normal traffic from accessing a network. An attack that tries to gather crucial information by sniffing a network is called a probe or surveillance. U2R is an attack that begins as a normal user account but in the future tries to gain access to the system as a super-user (root) to exploit all the access privileges. R2L is an attack that tries to gain local access to a remote machine.

The proposed schema for data preprocessing majorly consists of the OneHotEncoder (OHE), which transforms the categorical features into dummy numeric features. It is followed by min-max normalization techniques to ensure that all features are in the same range and the model is absolutely unbiased. All features are normalized in the range of [0, 1] using Equation (3). Here, min (z) and max (z) are the minimum and maximum values of any attribute Z. The original and normalized value of the feature are indicated by Z and $Z_{Normalized}$, respectively.

$$Z_{Normalized} = Z - \min(z)/\max(z) - \min(z) \quad (3)$$

The next important step is that of feature selection to shortlist the features that contribute most to the prediction variable, reduce the training time and prevent computational complexity. The proposed model is tested with three different Feature selection methodologies—Spark-Chi-SVM model, SVM classifier with a reduced set of input features. The Spark-Chi-SVM model combines the ChiSqSelector and SVM (support vector machines). The feature selection that is applied is the numTopFeatures method. The second technique first builds a classifier using all input features in the training dataset (M1). Then, it gradually removes one input feature and builds another classifier(M2). If the classification accuracy of M2 $\geq$ M1, then the algorithm considers the new set of features. The random forest technique checks the correlation among the features. It selects those features which affect other features and have high significance.

Different classifiers have been used to train and test the proposed IDS system such as NB, Logistic, MLP, SMO, IBK, J48. Only the selected features are trained. They are then analyzed on the basis of parameters such as Accuracy, Precision, Recall and F Score.

3.4. Corresponding Original Information Retrieval Scheme

The intended receiver can finally decrypt the Huffman Tree, cipher information and other communication essentials from the KryptosChain. It begins by traversing the entire cipher information characterwise from right to left. Simultaneously, the Huffman Tree is also referred to from the extreme rightmost node towards the leftmost nodes. Each time a Purine (A or G) is encountered, we go to the upper branch. The lower branch is referred to on hitting upon a Pyrimidine (T or C). By following this trend, once a symbol on the left-most node is reached, we immediately note it down. This process is followed until the entire cipher information string is traversed. The flowchart is given in Figure 9.

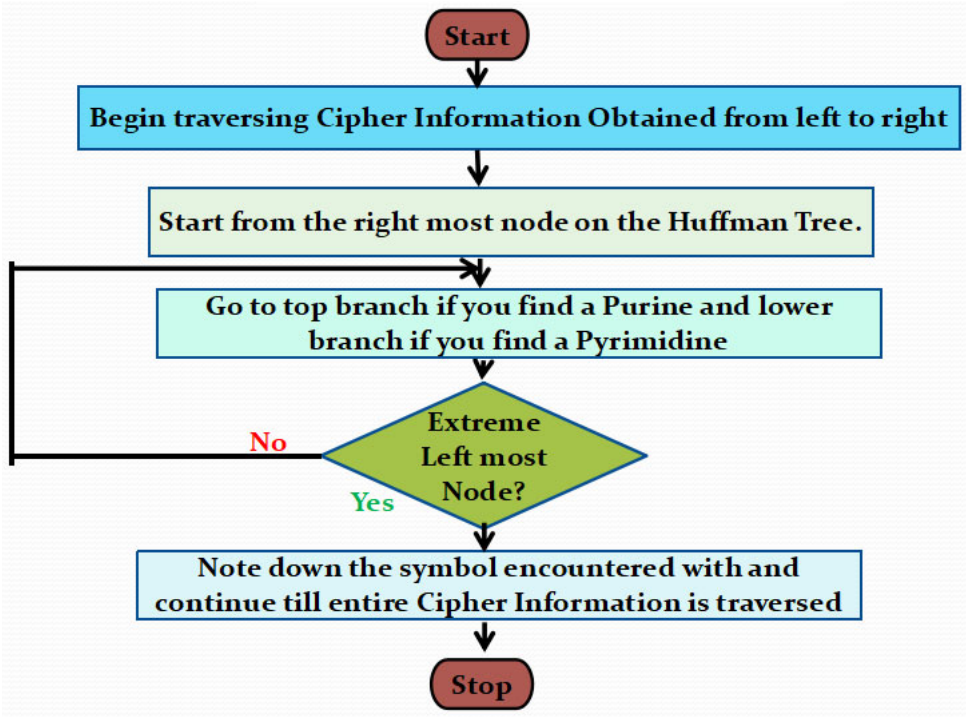

Figure 9. Flowchart of Original Information Retrieval Scheme.

4. Results and Calculations

This section provides the demonstration and calculations of the proposed scheme in detail.

4.1. Demonstration of Cipher Information Generation Scheme

Consider the Original Message to be: Cryptography is a method of protecting information and communications through the use of codes, so that only those for whom the information is intended can read and process it.

Frequency Distribution: C—1; r—9; y—3; p—4; t—15; o—19; g—3; a—10; h—9; i—11; s—9; m—6; e—11; d—7; f—5; c—6; n—13; u—3; ,—1; l—1; w—1; .—1

Alphabetical sequence categories symbols first, then lower case characters, and then lastly the upper case characters. Thus, rearranging in increasing order of frequency and in alphabetical sequence: ,—1; .—1; l—1; w—1; C—1; g—3; y—3; u—3; p—4; f—5; c—6; m—6; d—7; h—9; r—9; s—9; a—10; e—11; i—11; n—13; t—15; o—19

Figure 10 illustrates the detailed Huffman tree generation of each symbol in the Original Message and Table 4 gives the corresponding Huffman codes of those characters. The least two frequent symbols "," and "." have been merged into a new node labeled 2, which is the sum total of the frequencies of " ," and ".". Next, "," being the higher branch is denoted with and A, which is a Purine; and "." being the lower branch is denoted with T, which is the complimentary Pyrimidine of A. In the next round, again, all symbols are arranged in increasing order of frequency. This time "l" and "w" form a new node labeled as 2. Being the higher branch "l", this time it is denoted with another Purine G and correspondingly "w" is allocated with complementary pyrimidine C. The process continues till there are no unmerged symbols, and the last node is the sum total of the frequencies of all symbols. Next, to obtain the code, read from the last node till the concerned symbol and keep noting down the nitrogenous bases on the way from right to left.

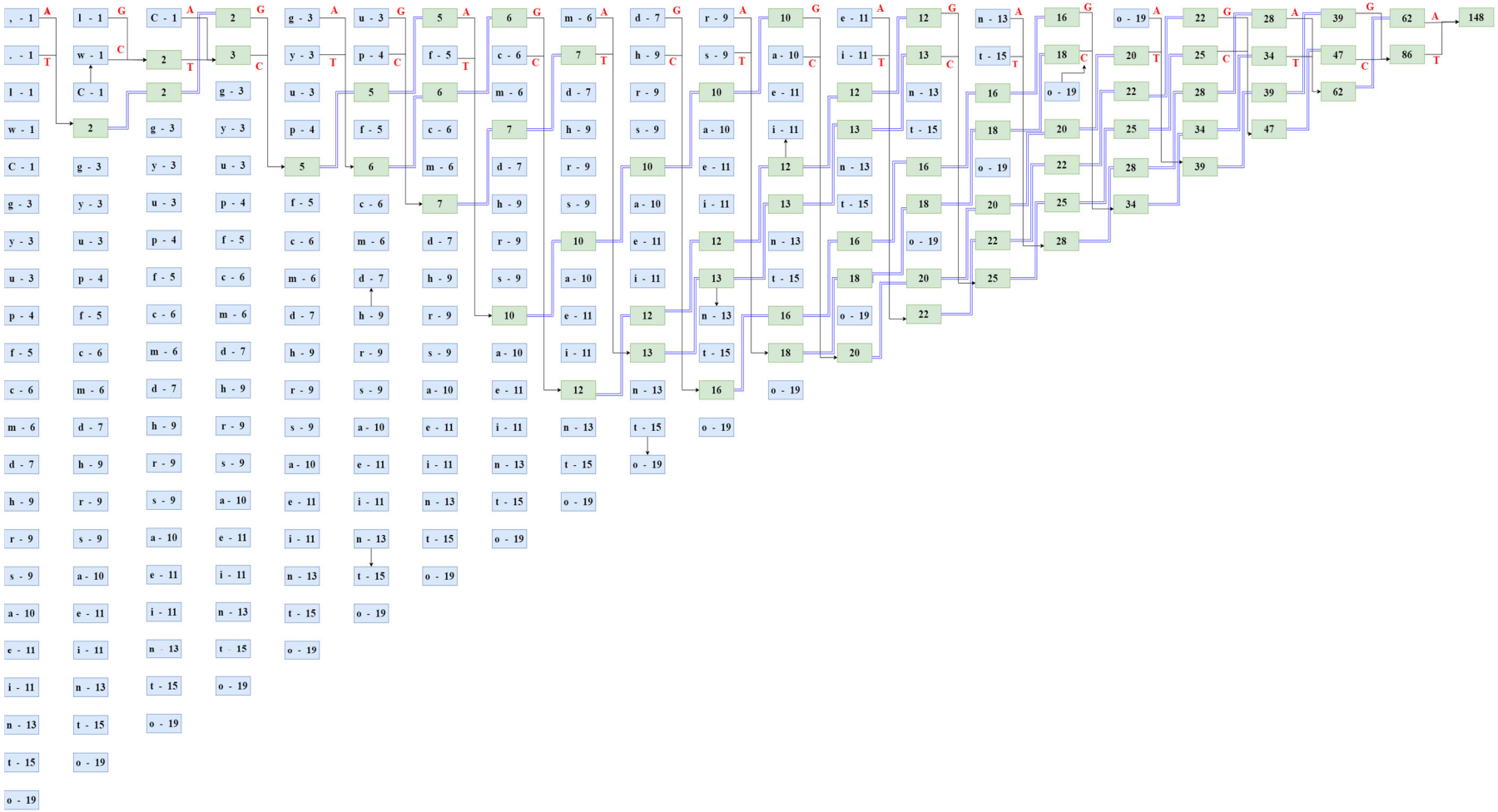

Figure 10. Demonstration of Huffman Tree Generation.

Table 4. Corresponding Huffman Codes for each symbol of the Original Message.

Character	Frequency	Huffman Code
,	1	TGTGAGA
.	1	TGTGAGT
l	1	TGTGACTG
w	1	TGTGACTC
C	1	TGTGACA
g	3	TCCGGA
y	3	TCCGGT
u	3	TCCCTG
p	4	TCCCTC
f	5	TGTGT
c	6	TCCGC
m	6	TCCCA
d	7	ATGG
h	9	ATGC
r	9	ATCA
s	9	TCTA
a	10	TGTC
e	11	TCGA
i	11	TCGT
I	13	AAA
t	15	AAT
o	19	TGA

The final cipher information will be obtained by replacing the original symbols with the Huffman codes—**TGTGACAATCATCCGGTTCCCTCAATTGATCCGGAATCATG TCTCCCTCATGCTCCGGTTCGTTCTATGTCTCCCATCGAAATATGCTGAATGGT GATGTGTTCCCTCATCATGAAATTCGATCCGCAATTCGTAAATCCGGATCGTAA ATGTGTTGAATCATCCCATGTCAATTCGTTGAAAATGTCAAAATGGTCCGCTGA TCCCATCCCATCCCTGAAATCGTTCCGCTGTCAATTCGTTGAAAATCTAAATAT GCATCATGATCCCTGTCCGGAATGCAATATGCTCGATCCCTGTCTATCGATGAT GTGTTCCGCTGAATGGTCGATCTATGTGAGATCTATGAAATATGCTGTCAATTG AAAATGTGACTGTCCGGTAATATGCTGATCTATCGATGTGTTGAATCATGTGAC TCATGCTGATCCCAAATATGCTCGATCGTAAATGTGTTGAATCATCCCATGTCA ATTCGTTGAAAATCGTTCTATCGTAAAAATTCGAAAAATGGTCGAATGGTCCG CTGTCAAAATCATCGATGTCATGGTGTCAAAATGGTCCCTCATCATGATCCGCT CGATCTATCTATCGT**

4.2. Demonstration of Cipher Information Transmission Scheme

The pictorial representation of all six phases is given from Figures 11–16 in a chronological order.

4.2.1. Phase 1: Registration Process

The public-private key pair is generated using the RSA scheme and then the hash of the public key is calculated by Alice using the SHA-256 algorithm. She next shares the hash of the public key to Kyrios, who stores it in the Look-up Table and returns a unique User ID to Alice as shown in Figure 11.

4.2.2. Phase 2: Sender Authentication

To authenticate the sender, Kyrios will match the hash of the public key sent by Alice with the value stored in his look-up table. Alice will be granted permission only if both the hashes match. This step is pictorially represented in Figure 12.

4.2.3. Phase 3: Genesis Block Generation Process

This phase is represented in Figure 13 along with the values of the different variables involved in this phase by Alice. The contents of the KryptosChain are shown after this phase is also depicted.

4.2.4. Phase 4: Receiver Authentication

The intended receiver of Alice is also similarly validated by Kyrios and this step is illustrated in Figure 14.

4.2.5. Phase 5: Second Block Generation Process

The detailed pictorial illustration of this step is presented in Figure 15, along with the view of the current contents of KryptosChain.

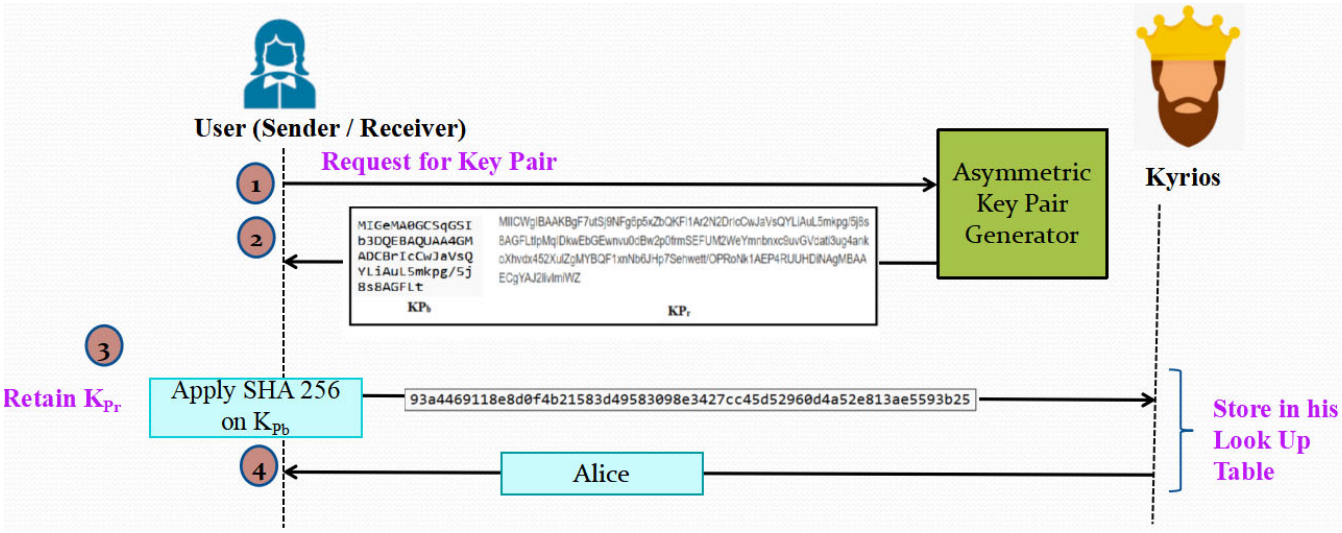

Figure 11. Demonstration of Phase 1: Registration Process.

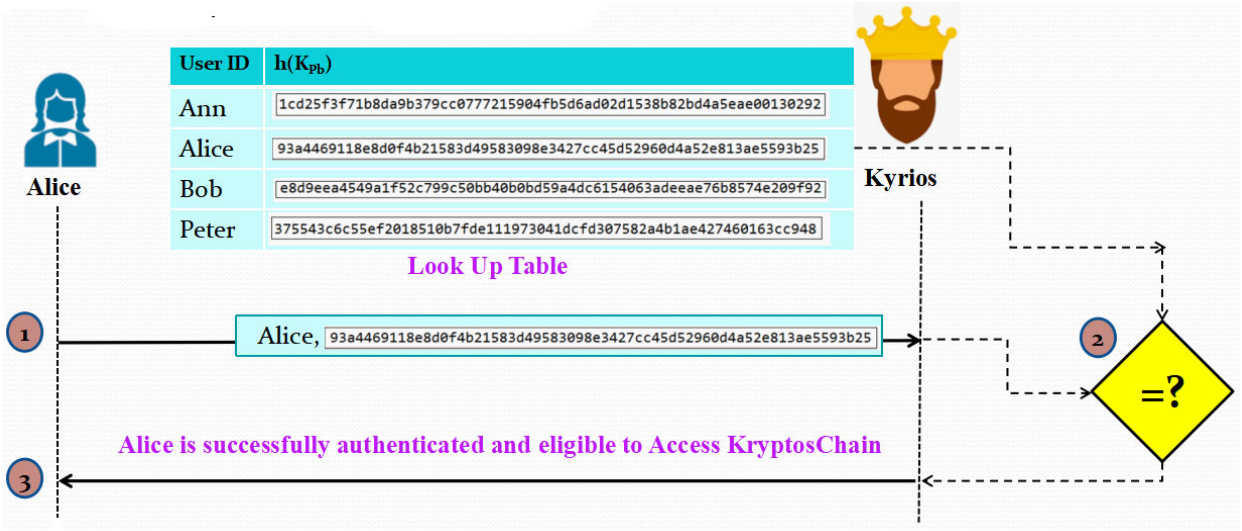

Figure 12. Demonstration of Phase 2: Sender Authentication.

4.2.6. Phase 6: Actual Information Exchange Process

Figure 16 gives the actual information exchange process. First, Alice and Bob establish the shared secret key and then continue communication by encrypting the additional contents of the blocks with it. It at this stage that the sender will share the encrypted block

containing the final ciphertext, Huffman tree—all important information related to key generation and selection with the receiver.

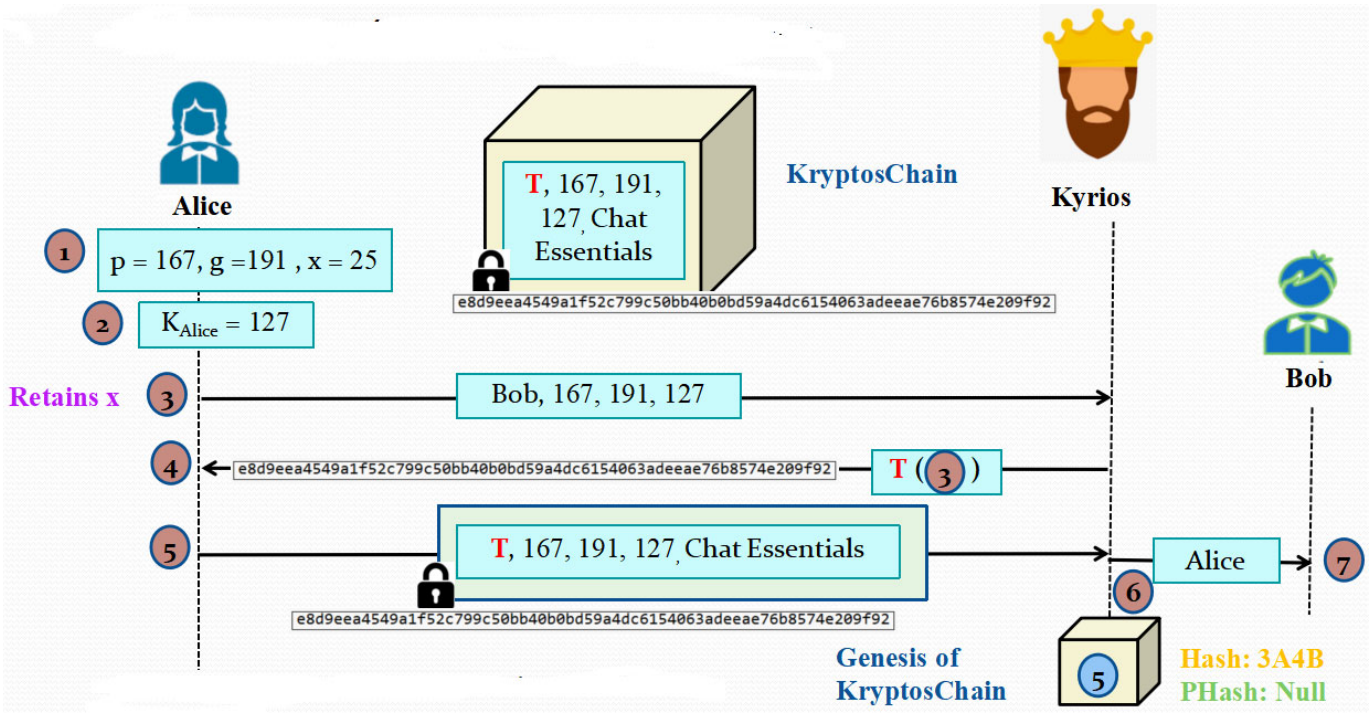

Figure 13. Demonstration of Phase 3: Genesis Block Generation Process.

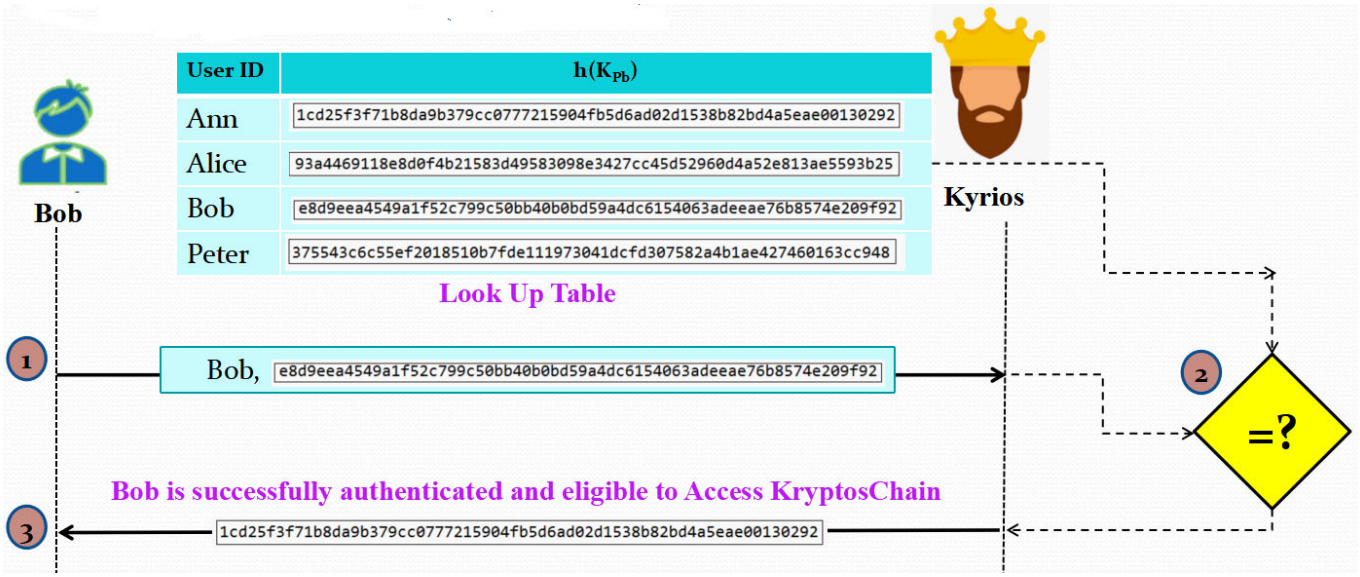

Figure 14. Demonstration of Phase 4: Receiver Authentication.

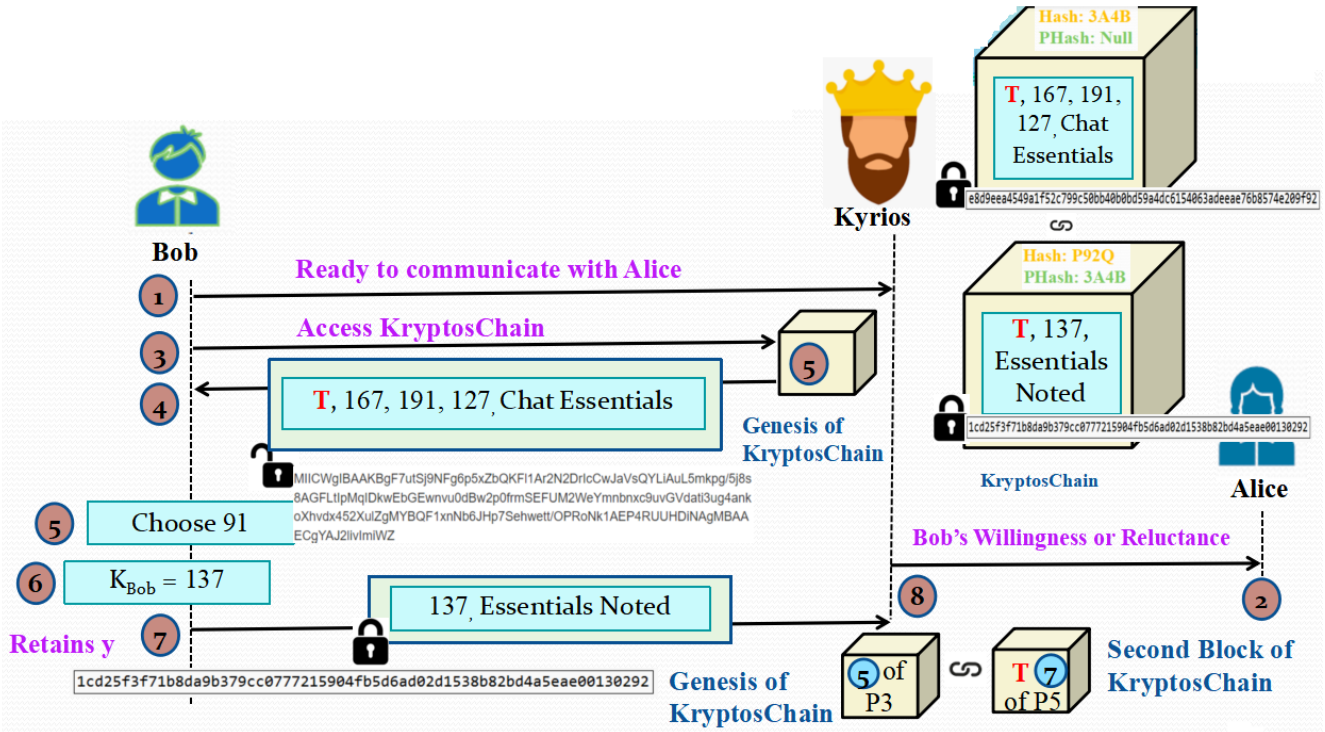

Figure 15. Demonstration of Phase 5: Second Block Generation Process.

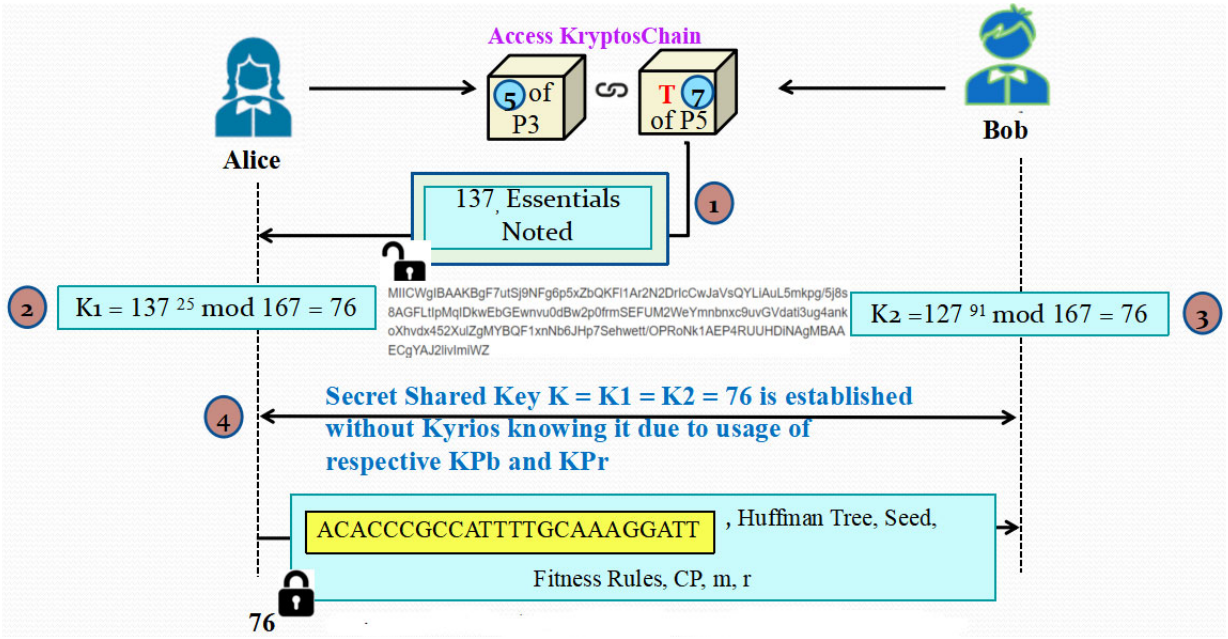

Figure 16. Demonstration of Phase 6: Actual Information Exchange Process.

4.3. Demonstration of Intrusion Detection System Scheme

The system configuration used for the implementation is InteI) I(TM) i5-8250U CPU@1.60 GHz. The observations related to the NSL KDD dataset is displayed below in Table 5.

The details of the NSL-KDD Training and Testing Dataset are further depicted in Tables 6 and 7

Table 5. Details of NSL_KDD dataset in general.

Parameter	Value	Further Specifications
Number of Features in the Dataset	41	Intrinsic Features 1–9 Content Features 10–22 Time-based Features 23–31 Host-based Features 32–41
Feature Type	4	Categorical (Features: 2, 3, 4, 42) Binary (Features: 7, 12, 14, 20, 21, 22) Discrete (Features: 8, 9, 15, 23–41, 43) Continuous (Features: 1, 5, 6, 10, 11, 13, 16, 17, 18, 19)
Number Of Attack Classes	22	DoS Attack Classes 6 Probe 4 R2L 8 U2R 4

Table 6. Details of NSL_KDD Training dataset.

Parameter	Value
Number of Rows	25,291
Number of Columns	42
Number of Missing Values	0
Number of Duplicate Records	0
Load Distribution	Normal-13488 Attack-11743

Table 7. Details of NSL_KDD Testing dataset.

Parameter	Value
Number of Rows	11,850
Number of Columns	42
Number of Missing Values	0
Number of Duplicate Records	0
Load Distribution	Normal-2152 Attack-9698

Thus, it is evident that there is no class imbalance in the training dataset as the number of normal and attack samples is almost equivalent.

The encoding demonstration of three of the categorical outputs—Protocol Type, Service and Flag feature of the NSL-KDD dataset whose original value is a string is shown in Table 8.

Table 8. Encoding of categorical features.

Protocol Type	Service	Flag	Protocol Type_Encoded	Service_Encoded	Flag_Encoded
icmp	http	SF	1	10	9
tcp	http	SF	2	10	9
tcp	ecr_i	SF	2	5	9
upd	private	SH	3	15	10
icmp	ecr_i	SF	1	5	9
udp	private	SH	3	15	10
tcp	http	SF	2	10	9
tcp	http	SF	2	10	9
tcp	http	SF	2	10	9
icmp	ecr_i	RSTR	1	5	2

An illustration of the normalization of the numeric values is shown in Table 9 using Equation (3).

Table 9. Normalization of Numeric Values.

src Bytes	dst Bytes	src Bytes_Normalized	dst Bytes_Normalized
1032	0	0	0
230	1041	0	0
336	406	0	0
30	0	0	0
378	838	0	0
54,550	8341	1	1
278	6845	0	0
0	0	0	0
1032	0	0	0
333	710	0	0

The accuracy obtained from different feature selection methodologies is depicted below in Table 10. Thus, the highest accuracy is achieved by considering only 17 features in the Spark-Chi-SVM model. The same accuracy is found by using all and only the 36 features in the SVM method.

Table 10. Accuracy of the applied Features Selection Techniques.

Spark-Chi-SVM Model		SVM with Reduced Features	
No. of Features	Accuracy (%)	No. of Features	Accuracy (%)
25	99.38	41	99.01
22	99.47	36	99.01
17	99.55	17	97.92
15	99.38	9	95.48
11	92.35	3	91.01

Different classifiers are used to train and test the dataset and the analysis is shown in Section 5.3. For that, the following metrics have been defined:

- **True Positive (TP)**—Attack data that is correctly classified as an attack.
- **False Positive (FP)**—Normal data that is incorrectly classified as an attack.
- **True Negative (TN)**—Normal data that is correctly classified as normal.
- **False Negative (FN)**—Attack data that is incorrectly classified as normal.

The following performance evaluation matrices are used to analyze the working of the classifiers.

Accuracy, which measures the proportion of the total number of correct classifications, is given by Equation (4).

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{4}$$

Precision gives the number of correct classifications penalized by the number of incorrect classifications as shown in Equation (5).

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{5}$$

Recall counts the number of correct classifications penalized by the number of missed entries as depicted in Equation (6).

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{6}$$

F-score measures the harmonic mean of precision and recall as given by Equation (7).

$$\mathrm{F-score} = \frac{2 \times \ \mathrm{Precision} \times \mathrm{Recall}}{\mathrm{Precision} \ + \ \mathrm{Recall}} \tag{7}$$

Sensitivity gives the number of intrusions detected correctly

$$\mathrm{Sensivity} = \mathrm{TP} \ + \ \mathrm{TP} \ \times \ \mathrm{FN} \ \times 100 \tag{8}$$

4.4. Demonstration of Original Information Retrieval Scheme

The intended receiver retrieves the following cipher message **TGTGACAATCATC CGGTTCCCTCAATTGATCCGGAATCATGTCTCCCTCATGCTCCGGTTCGTTCTA TGTCTCCCATCGAAATATGCTGAATGGTGATGTGTTCCCTCATCATGAAATTCG ATCCGCAATTCGTAAATCCGGATCGTAAATGTGTTGAATCATCCCATGTCAATT CGTTGAAAATGTCAAAATGGTCCGCTGATCCCATCCCATCCCTGAAATCGTTC CGCTGTCAATTCGTTGAAAATCTAAATATGCATCATGATCCCTGTCCGGAATG CAATATGCTCGATCCCTGTCTATCGATGATGTGTTCCGCTGAATGGTCGATCTA TGTGAGATCTATGAAATATGCTGTCAATTGAAAATGTGACTGTCCGGTAATATG CTGATCTATCGATGTGTTGAATCATGTGACTCATGCTGATCCCAAATATGCTCG ATCGTAAATGTGTTGAATCATCCCATGTCAATTCGTTGAAAATCGTTCTATCGT AAAAATTCGAAAAATGGTCGAATGGTCCGCTGTCAAAATCATCGATGTCATGG TGTCAAAATGGTCCCTCATCATGATCCGCTCGATCTATCTATCGT.**

The Huffman tree is already depicted in Figure 10. We begin from the rightmost node labeled 148. The first character of the cipher message is T (i.e., a Pyrimidine) so we go to the lower branch 86. The next character is G (i.e., a Purine) so we go to the higher branch 39. Next, we trace across two boxes labeled 39 using the blue arrows to find that box of 39, which is formed by adding to lower-valued boxes. The next character is T (i.e., a Pyrimidine) so we go to the lower branch 20. Again, we trace across four boxes labeled as 20. The next character is G (i.e., a Purine) so we go to the higher branch 10, and then trace across four 10 boxes. The next character is A (i.e., a Purine) so we go to the higher branch 5 and trace two more 5 boxes. The next character is C (i.e., a Pyrimidine) so we go to the lower branch 3. The next character is A so we go to the higher branch and encounter the symbol "C". Thus, the first symbol of the original message is successfully retrieved. We follow similar steps to retrieve the entire original message.

5. Analysis of Proposed Model

This section provides the meticulous analysis of each proposed scheme in terms of the parameters mentioned in the subsections, as well as their comparison with existing similar models schemes

5.1. Analysis of Proposed Cipher Information Generation Scheme

5.1.1. Number of Bits Required to Encode

Table 11 showcases the comparison of number of bits required to encode each symbol of the considered original message using ASCII encoding, traditional DNA encoding, wherein each symbol is represented as a codon using a pre-decided table (shown in Table 1) and proposed DNA-based Huffman encoding.

It thus is evident that ASCII encoding needs a huge number of bits to encode the same symbols as compared with traditional DNA code and the DNA Huffman code, thus giving large-space complexity. Traditional DNA codes need a number of bits in comparison with DNA Huffman codes but since their length is fixed, they are more guessable and prone to brute force attacks. The variable length of the DNA Huffman codes make them immune from intrusions. Thus, it is evident that the proposed method is better in terms of code length and security.

Table 11. Analysis on Numbers of bits required to Encode.

Character	ASCII Code	Traditional DNA Code	Proposed DNA Huffman Code
,	00101100	CCC	TGTGAGA
.	00101110	GTC	TGTGAGT
l	01101100	TGC	TGTGACTG
w	01110111	CCA	TGTGACTC
C	01000011	GAC	TGTGACA
g	01100111	ATG	TCCGGA
y	01111001	AAA	TCCGGT
u	01110101	ATA	TCCCTG
p	01110000	GTC	TCCCTC
f	01100110	AGT	TGTGT
c	01100011	GAC	TCCGC
m	01101101	TCA	TCCCA
d	01100100	GAT	ATGG
h	01101000	CGT	ATGC
r	01110010	CAC	ATCA
s	01110011	AGT	TCTA
a	01100001	CTG	TGTC
e	01100101	GCG	TCGA
i	01101001	AAG	TCGT
n	01101110	GCT	AAA
t	01110100	TTA	AAT
o	01101111	GAA	TGA
Total number of bits	176 Bits	66 Bits	113 bits

5.1.2. Effect of Increase in Number of Symbols in Plaintext

Figure 17 shows the effect of an increase in the number of symbols on the numbers of bits required to represent in ASCII encoding as well as the proposed scheme.

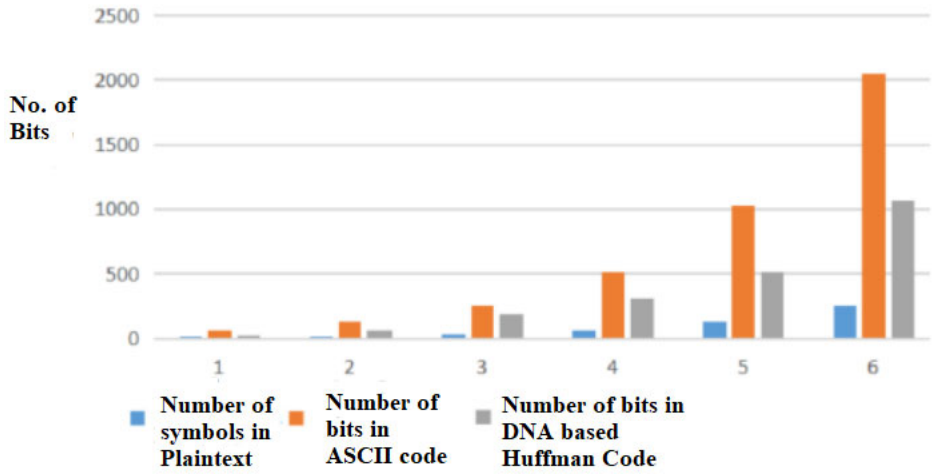

Figure 17. Comparison of number of bits needed in ASCII code and DNA-based Huffman code as Number of Symbols in Plaintext is doubled.

It is evident that the number of bits needed in ASCII is huge in contrast to the number of bits needed in the proposed DNA encoding scheme when the number of symbols is doubled.

5.1.3. Security Analysis

Usage of variable length makes the codes less guessable and shows strong immunity against brute force attacks as the intruder cannot guess the probable key space.

5.1.4. Complexity Analysis

The Total time to calculate the DNA encoded string for all symbols depends on the total number of distinct symbols "n". It uses a heap to store the weight of each tree; each iteration requires O(log n) time to determine the cheapest weight and insert the new weight. There are O(n) iterations, one for each item. Thus, the Time Complexity is of the order O(n log n).

For Space Complexity, at max there can be O(n) extra nodes so n nodes for characters and O(n) extra nodes, which in total is still O(n).

5.1.5. Comparison of Proposed Model with Existing Similar Models

The comparison of the proposed cipher generation scheme with some of the existing similar works is portrayed in Table 12.

Table 12. Comparison of Proposed Cipher Generation Scheme with Existing Similar Models.

Parameter	Smith et al. [53]	Aeilenberg and Rotstein [54]	Meftah et al. [55]	Proposed Scheme
Frequency Distribution Value	Standard frequency of English alphabets	Standard frequency of English alphabets	Probability of appearance of a short Sequence in the initially chosen DNA string	Real-time occurrence of symbols in the considered plaintext
Type of Plaintext	Only English alphabets	All letters, numbers, characters	Input Image	All letters, numbers, characters
Resultant Huffman code for				
,	, = No Code	, = TAAT	No methodology described for text symbols	, = TGTGAGA
.	. = No Code	. = TCTA		. = TGTGAGT
l	l = AAA	l = TTAG		l = TGTGACTG
w	w = AAT	w = TATG		w = TGTGACTC
C	C = AAG	C = TTCG		C = TGTGACA
g	g = ACT	g = GAAT		g = TCCGGA
y	y = ACC	y = TACG		y = TCCGGT
u	u = AAC	u = TACT		u = TCCCTG
p	p = CCA	p = GAAC		p = TCCCTC
f	f = ACG	f = TAAG		f = TGTGT
c	c = AAG	c = TTCG		c = TCCGC
m	m = ACA	m = TAAC		m = TCCCA
d	d = CT	d = TTAC		d = ATGG
h	h = CA	h = TTTC		h = ATGC
r	r = CG	r = TTTG		r = ATCA
s	s = GT	s = TTCT		s = TCTA
a	a = AT	a = GAC		a = TGTC
e	e = T	e = GCT		e = TCGA
i	i = GG	i = GCG		i = TCGT
n	n = GC	n = TTAT		n = AAA
t	t = AG	t = GTG		t = AAT
o	o = GA	o = GAG		o = TGA
As described by authors				
Re-usability of the codes?	Yes, as same Huffman tree is generated every time as frequency of characters remain the same.	Yes, as same Huffman tree is generated every time as frequency of characters remain the same.	No, as they are based on tedious biological processes.	No, as it considers real-time frequency of occurrence of the symbols involved.
Case sensitive?	No	No	Not applicable	Yes

It is evident from Table 12 that most of the similar existing works rely on standard English frequency, thus giving a fixed Huffman tree, which can be readily constructed at the intruder site because even he can access the frequencies of the alphabets. Some of the existing schemes consider only alphabets as their plaintext. In contrast to already proposed similar models, our proposed model considers all alphabets, numbers and characters as input. It always considers the real-time frequency of the plaintext symbols, which mandates the calculation of a fresh Huffman tree each time. Although it is time-consuming, security wise it is better as it is less prone to intrusion by adversaries. Other schemes are case insensitive. However, the proposed scheme provides distinction between the lower and upper case letters.

5.2. Analysis of the Proposed Cipher Transmission Generation Scheme

5.2.1. Attainment of Principles of Security

- **Achieving Confidentiality:** The proposed KryptosChain achieves confidentiality as any adversary or even the trusted third party Kyrios only get to see either the hash of the public key or encrypted contents locked with the hash of the public key or shared secret key. Hashes are one-way so the original values cannot be obtained from them. If any content is encrypted with the hash of the public key, then only a corresponding hash of the private key can unlock it. Private keys are always kept as a secret and never revealed to the outer world. The shared secret key is calculated at the respective ends of the sender and receiver and never transmitted directly through the KryptosChain.
- **Achieving Integrity:** The fundamental feature of the blockchain stores the hash of the contents of the previous block into its successive blocks. KryptosChain also employs this basic feature. If any block is tampered with, the hash automatically alters. This will lead to a hash mismatch with the hash stored in the successive block. This helps to easily identify any contaminated blockchain and assure integrity.
- **Achieving Availability:**All users after successful registration can access KryptosChain whenever needed, thus providing availability.
- **Achieving Authenticity:**The responsibility to authenticate each user is bestowed upon Kyrios, which validates them by referring to the look-up table.
- **Achieving Non-Repudiation:** If anything is uploaded once into KryptosChain, it is immutable; thus, there is no repudiation possible at a later stage by any user. The addition of timestamps by Kyrios also eliminates any kind of refusals in the future.
- **Achieving Access Control:** Even if in the rarest of cases an adversary also successfully registers himself, he too cannot comprehend anything. The reason is that each and every content in the KryptosChain is encrypted and in an unreadable form. Thus, the proposed scheme eliminates any chances of man-in-the-middle attacks.

Thus, the summarized attainment of the principles of security is depicted in Table 13.

Table 13. Attainment of Principles of Security by Proposed Cipher Transmission Scheme.

Goal.	Status	Justification
Confidentiality	☑	Kyrios or a registered adversary cannot read the contents of the blocks as they are all encrypted.
Integrity	☑	Hash mismatch denotes any kind of tampering.
Availability	☑	All successfully registered users can access KryptosChain whenever needed.
Authentication	☑	Kyrios authenticates each user by referring to his look-up table and uses timestamps.
Non-Repudiation	☑	Basic immutability of KryptosChain and timestamps refute repudiations in the future.
Access Control	☑	Unwanted parties can be debarred from access by Kyrios and encryption resists man-in-the -middle attack.

5.2.2. Immunity to Cryptographic Attacks

The summarized attainment of immunity to common cryptographics is depicted in Table 14.

Table 14. Attainment of Immunity to Cryptographic Attacks by Proposed Cipher Transmission Scheme.

Attack	Immunity	Justification
Ciphertext Only Attack	☑	The block containing the final Ciphertext, Huffman tree and other information is encrypted by the shared secret key established secretively between Alice and Bob through KryptosChain. Even Kyrios is not aware of this key.
Known Plaintext Attack	☑	The final DNA key is chosen from a pool of best DNA keys and changed for each encryption process.
Chosen Plaintext Attack	☑	The intruder needs to undergo the registration phase only; then, he gets access to KryptosChain and encryption machinery.
Chosen Ciphertext Attack	☑	The intruder needs to undergo the registration phase only; then, he gets access to KryptosChain and decryption machinery.
Replay Attack	☑	The intruder needs to undergo the registration phase only; then, he gets access to KryptosChain.
Side Channel Attack	☑	All the information is encrypted in the form of a chain of blocks.
Brute Force Attack	☑	All efforts are futile as everything is encrypted with suitable asymmetric keys, which are difficult to guess.

5.2.3. Comparison of Proposed Model with Existing Similar Models

The comparison of the proposed cipher transmission scheme with some of the existing similar works is portrayed in Table 15.

Table 15. Comparison of Proposed Cipher Transmission Scheme with Existing Similar Models.

Parameter	Menegay et al. [34]	Naz et al. [35]	Ellewala et al. [37]	Singh et al. [38]	Khacef & Pujjole [39]	Proposed Scheme
Real Blockchain usage	Yes	Yes	Yes	Yes	Yes	No
Type of Blockchain	Public	Public with IPFS and smart contracts	Private with encryption	Public with digital certificates	Public with PKI	Not applicable
Crypto Currency	Yes	Yes	Yes	Yes	Yes	No
Comments	Email server added in an existing Blockchain	Digital assests are shared and delivered	Restricted to a single enterprise	Each user uploads his public key certificate into the Blockchain	Instead of CA Blockchain enables distribution of keys	A Blockchain-inspired Diffie Hellman protocol is used
Limitations	Scalability issues as number of users increases	Economically cumbersome	Private blockchains are expensive	Highly dependent on MNO to provide certificates	Scalability issues as number of users increase	6 phases need to be passed

5.3. Analysis of Proposed Intrusion Detection Scheme

Table 16 represents the performance evaluations of various classifiers applied on the proposed IDS system, which puts forward that J48 is the best classifier.

Table 16. Classifiers Training vs. Testing Performance Evaluation.

Classifier	Accuracy (%)	Precision	Recall	F-Score	Time to Train (msec)	Sensitivity (%)
NB	78.35	0.812	0.785	0.798	2457	75.45
Logistics	82.47	0.856	0.815	0.836	2490	83.14
MLP	80.35	0.808	0.785	796	2503	79.89
SMO	85.56	0.832	0.798	0.814	2734	84.67
IBK	91.35	0.837	0.807	0.821	2556	91.28
J48	95.84	0.868	0.838	0.852	2769	96.78

It is thus evident that the J48 classifier provides the highest Accuracy, Precision, Recall, F-Score and Sensitivity compared with its counterparts. Time to train is, however, high for J48.

6. Conclusions and Future Work

The proposed scheme uses Huffman coding fundamentals due its feature of assigning variable codes and smaller codes for more frequently occurring symbols. It considers the real-time occurrence of every distinct symbol in the plaintext to determine their frequency distribution. In contrast to assigning a 1 to the higher branch and 0 to the lower branch after adding the two least frequent symbols, the proposed scheme alternatively assigns a purine and pyrimidine value to the high and low branch. A different Huffman tree needs to be derived each time to get the corresponding codes, thus enhancing the security. The variable length of the codes also make them less guessable and immune to attacks.

To transmit the ciphertext to the intended receiver, a Blockchain-inspired scheme has been proffered. Real Blockchains necessitate possession of cryptocurrencies, writing smart contracts, deploying them and arriving at consensus to get the enormous facilities. Therefore, this paper produces a blockchain-inspired KryptosChain scheme that transmits fixed-sized blocks of the original message to the genuine receiver. All the essential goals of security are attained by the proposed KryptosChain scheme as it stores the hash of the current block into its successive block. The trusted third party Kyrios is only involved to authenticate the sender and receiver and unknowingly assist them to establish the shared secret key. Kyrios only knows the hash of the public key, and he cannot obtain the actual public key as hashes are one-way The actual message exchange is also safeguarded from Kyrios as they are encrypted by the intended receiver's public key, which only he can decrypt with a corresponding private key. Thus, involved parties can exchange information securely via KryptosChain.

A panacea to any further possible intrusion on KryptosChain is curbed by an AI-based IDS system. Various classifiers, namely-NB (naïve Bayes), logistics, MLP (multi-layer perceptron), SMO, IBK, and J48 were employed for classification of data as normal or attack on the NSL-KDD dataset. Empirical results reveal that the performance of J48 was admirable at 95.84%. Thus, the proposed model has an effective prediction rate, and also reduces the computational complexity by removing irrelevant features using appropriate pre-processing and feature selection methodologies.

Author Contributions: Data curation, P.M.; Formal analysis, C.P., H.K.T. and P.M.; Investigation, P.M., T.G. and C.P.; Methodology, P.M. and C.P.; Project administration, H.K.T. and T.G.; Resources, H.K.T. and T.G.; Software, P.M. and C.P.; Validation, C.P., H.K.T. and T.G.; Visualization, H.K.T. and T.G.; Writing—original draft, P.M. and C.P.; Writing—review and editing, H.K.T. and T.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Cui, G.; Qin, L.; Wang, Y.; Zhang, X. An encryption scheme using DNA technology. In Proceedings of the 2008 3rd International Conference on Bio-Inspired Computing: Theories and Applications, Beijing, China, 2–4 November 2018; pp. 37–42.
2. Mondal, M.; Ray, K.S. Review on DNA cryptography. *arXiv* **2019**, arXiv:1904.05528.
3. Mukherjee, P.; Garg, H.; Pradhan, C.; Ghosh, S.; Chowdhury, S.; Srivastava, G. Best Fit DNA-Based Cryptographic Keys: The Genetic Algorithm Approach. *Sensors* **2022**, *22*, 7332. [CrossRef] [PubMed]
4. Kumar, N.M.; Mallick, P.K. Blockchain technology for security issues and challenges in IoT. *Procedia Comput. Sci.* **2018**, *132*, 1815–1823. [CrossRef]
5. Mukherjee, P.; Pradhan, C. Blockchain 1.0 to blockchain 4.0—The evolutionary transformation of blockchain technology. In *Blockchain Technology: Applications and Challenges*; Springer: Cham, Switzerland, 2021; pp. 29–49.
6. Mukherjee, P.; Barik, R.K.; Pradhan, C. A comprehensive proposal for blockchain-oriented smart city. In *Security and Privacy Applications for Smart City Development*; Springer: Cham, Switzerland, 2021; pp. 55–87.
7. Mukherjee, P.; Barik, L.; Pradhan, C.; Patra, S.S.; Barik, R.K. hQChain: Leveraging Towards Blockchain and Queueing Model for Secure Smart Connected Health. *Int. J. E-Health Med. Commun.* **2021**, *12*, 1–20. [CrossRef]
8. Mukherjee, P.; Barik, R.K.; Pradhan, C. Agrochain: Ascending blockchain technology towards smart agriculture. In *Advances in Systems, Control and Automations*; Springer: Singapore, 2021; pp. 53–60.
9. Mukherjee, P.; Barik, R.K.; Pradhan, C. eChain: Leveraging Toward Blockchain Technology for Smart Energy Utilization. In *Applications of Advanced Computing in Systems*; Springer: Singapore, 2021; pp. 73–81.
10. Mohamed, T.; Gaber, T.; Goda, E.; Snasel, V.; Ella Hassanien, A. A Blockchain Protocol for Authenticating Space Communications between Satellites Constellations. *Aerospace* **2022**, *9*, 495. [CrossRef]
11. Ma, Y.; Sun, Y.; Lei, Y.; Qin, N.; Lu, J. A survey of blockchain technology on security, privacy, and trust in crowdsourcing services. *World Wide Web* **2020**, *23*, 393–419. [CrossRef]
12. Mohanta, B.K.; Jena, D.; Panda, S.S.; Sobhanayak, S. Blockchain technology: A survey on applications and security privacy challenges. *Internet Things* **2019**, *8*, 100107. [CrossRef]
13. Zheng, Z.; Xie, S.; Dai, H.; Chen, X.; Wang, H. An overview of blockchain technology: Architecture, consensus, and future trends. In Proceedings of the 2017 IEEE International Congress on Big Data (BigData Congress), Boston, MA, USA, 25–30 June 2017; pp. 557–564.
14. Watanabe, H.; Fujimura, S.; Nakadaira, A.; Miyazaki, Y.; Akutsu, A.; Kishigami, J.J. Blockchain contract: A complete consensus using blockchain. In Proceedings of the 2015 IEEE 4th Global Conference on Consumer Electronics (GCCE), Osaka, Japan, 27–30 October 2015; pp. 577–578.
15. Wirkuttis, N.; Klein, H. Artificial intelligence in cybersecurity. *Cyber Intell. Secur.* **2017**, *1*, 103–119.
16. Corea, F. The convergence of AI and blockchain. In *Applied Artificial Intelligence: Where AI Can Be Used in Business*; Springer: Cham, Switzerland, 2019; pp. 19–26.
17. Zolfaghari, B.; Koshiba, T. AI Makes Crypto Evolve. *Appl. Syst. Innov.* **2022**, *5*, 75. [CrossRef]
18. Zolfaghari, B.; Rabieinejad, E.; Yazdinejad, A.; Parizi, R.M.; Dehghantanha, A. Crypto Makes AI Evolve. *arXiv* **2022**, arXiv:2206.12669. [CrossRef]
19. Vitter, J.S. Design and analysis of dynamic Huffman codes. *J. ACM* **1987**, *34*, 825–845. [CrossRef]
20. Li, N. Research on Diffie-Hellman key exchange protocol. In Proceedings of the 2010 2nd International Conference on Computer Engineering and Technology, Chengdu, China, 16–19 April 2010; Volume 4, pp. 634–637.
21. Jain, S.; Bhatnagar, V. A novel DNA sequence dictionary method for securing data in DNA using spiral approach and framework of DNA cryptography. In Proceedings of the 2014 International Conference on Advances in Engineering & Technology Research, Kanpur, India, 1–2 August 2014; pp. 1–5.
22. Hameed, S.M.; Sa'adoon, H.A.; Al-Ani, M. Image encryption using DNA encoding and RC4 algorithm. *Iraqi J. Sci.* **2018**, 434–446.
23. Nandy, N.; Banerjee, D.; Pradhan, C. Color image encryption using DNA based cryptography. *Int. J. Inf. Technol.* **2021**, *13*, 533–540. [CrossRef]
24. Ning, K. A pseudo DNA cryptography method. *arXiv* **2009**, arXiv:0903.2693.
25. Dhawan, S.; Saini, A. Integration of DNA Cryptography for Complex Biological Interactions. Available online: https://www.academia.edu/en/28702512/Integration_of_DNA_Cryptography_for_Complex_Biological_Interactions (accessed on 29 December 2022).
26. Zhang, Y.; Fu, B.; Zhang, X. DNA cryptography based on DNA Fragment assembly. In Proceedings of the 2012 8th International Conference on Information Science and Digital Content Technology (ICIDT2012), Jeju Island, Republic of Korea, 26–28 June 2012; Volume 1, pp. 179–182.
27. Zhang, Y.; Wang, Z.; Wang, Z.; Karanfil, Y.H.; Dai, W. A new DNA cryptography algorithm based on the biological puzzle and DNA chip techniques. In *International Conference on Biomedical and Biological Engineering*; Atlantis Press: Amsterdam, The Netherlands, 2016; pp. 360–365.
28. Wang, Y.; Han, Q.; Cui, G.; Sun, J. Hiding messages based on DNA sequence and recombinant DNA technique. *IEEE Trans. Nanotechnol.* **2019**, *18*, 299–307. [CrossRef]
29. Singh, M.S.P.; Naidu, M.E. A Novel method to secure data using DNA sequence and Armstrong Number. *Asian J. Converg. Technol.* **2017**, *3*, 40.

30. Sukumaran, S.C.; Misbahuddin, M. DNA Cryptography for Secure Data Storage in Cloud. *Int. J. Netw. Secur.* **2018**, *20*, 447–454.
31. Pujari, S.K.; Bhattacharjee, G.; Bhoi, S. A hybridized model for image encryption through genetic algorithm and DNA sequence. *Procedia Comput. Sci.* **2018**, *125*, 165–171. [CrossRef]
32. Partala, J. Provably secure covert communication on blockchain. *Cryptography* **2018**, *2*, 18. [CrossRef]
33. Guziur, J.; Pawlak, M.; Poniszewska-Marańda, A.; Wieczorek, B. Light blockchain communication protocol for secure data transfer integrity. In *International Symposium on Cyberspace Safety and Security*; Springer: Cham, Switzerland, 2018; pp. 194–208.
34. Sarıtekin, R.A.; Karabacak, E.; Durgay, Z.; Karaarslan, E. Blockchain based secure communication application proposal: Cryptouch. In Proceedings of the 2018 6th International Symposium on Digital Forensic and Security (ISDFS), Antalya, Turkey, 22–25 March 2018; pp. 1–4.
35. Menegay, P.; Salyers, J.; College, G. Secure communications using blockchain technology. In Proceedings of the MILCOM 2018-2018 IEEE Military Communications Conference (MILCOM), Los Angeles, CA, USA, 29–31 October 2018; pp. 599–604.
36. Naz, M.; Al-zahrani, F.A.; Khalid, R.; Javaid, N.; Qamar, A.M.; Afzal, M.K.; Shafiq, M. A secure data sharing platform using blockchain and interplanetary file system. *Sustainability* **2019**, *11*, 7054. [CrossRef]
37. Bi, W.; Yang, H.; Zheng, M. An accelerated method for message propagation in blockchain networks. *arXiv* **2018**, arXiv:1809.00455.
38. Ellewala, U.P.; Amarasena, W.D.H.U.; Lakmali, H.S.; Senanayaka, L.M.K.; Senarathne, A.N. Secure Messaging Platform Based on Blockchain. In Proceedings of the 2020 2nd International Conference on Advancements in Computing (ICAC), Colombo, Sri Lanka, 11 December 2020; Volume 1, pp. 317–322.
39. Singh, R.; Chauhan, A.N.S.; Tewari, H. Blockchain-enabled end-to-end encryption for instant messaging applications. In Proceedings of the 2022 IEEE 23rd International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), Belfast, UK, 14–17 June 2022; pp. 501–506.
40. Khacef, K.; Pujolle, G. March. Secure Peer-to-Peer communication based on Blockchain. In *Workshops of the International Conference on Advanced Information Networking and Applications*; Springer: Cham, Switzerland, 2019; pp. 662–672.
41. Zhang, B.; Zhang, T.; Yu, Z. DDoS detection and prevention based on artificial intelligence techniques. In Proceedings of the 2017 3rd IEEE International Conference on Computer and Communications (ICCC), Chengdu, China, 13–16 December 2017; pp. 1276–1280.
42. Glăvan, D.; Răcuciu, C.; Moinescu, R.; Antonie, N.F. DDoS detection and prevention based on artificial intelligence techniques. *Sci. Bull. Mircea Cel Batran Nav. Acad.* **2019**, *22*, 1–11.
43. Jaszcz, A.; Połap, D. AIMM: Artificial Intelligence Merged Methods for flood DDoS attacks detection. *J. King Saud Univ. -Comput. Inf. Sci.* **2022**, *34*, 8090–8101. [CrossRef]
44. Repalle, S.A.; Kolluru, V.R. Intrusion detection system using ai and machine learning algorithm. *Int. Res. J. Eng. Technol.* **2017**, *4*, 1709–1715.
45. Kim, A.; Park, M.; Lee, D.H. AI-IDS: Application of deep learning to real-time Web intrusion detection. *IEEE Access* **2020**, *8*, 70245–70261. [CrossRef]
46. Gaber, T.; El-Ghamry, A.; Ella Hassanien, A. Injection attack detection using machine learning for smart IoT applications. *Phys. Commun.* **2022**, *52*. [CrossRef]
47. Majid, A.A.M.; Alshaibi, A.J.; Kostyuchenko, E.; Shelupanov, A. A review of artificial intelligence based malware detection using deep learning. *Mater. Today Proc.* **2021**. [CrossRef]
48. Faruk, M.J.H.; Shahriar, H.; Valero, M.; Barsha, F.L.; Sobhan, S.; Khan, M.A.; Whitman, M.; Cuzzocrea, A.; Lo, D.; Rahman, A.; et al. Malware detection and prevention using artificial intelligence techniques. In Proceedings of the 2021 IEEE International Conference on Big Data (Big Data), Orlando, FL, USA, 15–18 December 2021; pp. 5369–5377.
49. Marwala, T.; Xing, B. Blockchain and artificial intelligence. *arXiv* **2018**, arXiv:1802.04451.
50. Zheng, Z.; Xie, S.; Dai, H.N.; Chen, W.; Chen, X.; Weng, J.; Imran, M. An overview on smart contracts: Challenges, advances and platforms. *Future Gener. Comput. Syst.* **2020**, *105*, 475–491. [CrossRef]
51. KDD Cup 1999. Available online: http://Kdd.Ics.Uci.Edu/Databases/Kddcup99.html (accessed on 18 August 2014).
52. NSL-KDD Dataset. Available online: http://nsl.cs.unb.ca/nsl-kdd/ (accessed on 21 July 2014).
53. Smith, G.C.; Fiddes, C.C.; Hawkins, J.P.; Cox, J.P. Some possible codes for encrypting data in DNA. *Biotechnol. Lett.* **2003**, *25*, 1125–1130. [CrossRef]
54. Ailenberg, M.; Rotstein, O.D. An improved Huffman coding method for archiving text, images, and music characters in DNA. *Biotechniques* **2009**, *47*, 747–754. [CrossRef] [PubMed]
55. Meftah, M.; Pacha, A.A.; Hadj-Said, N. DNA encryption algorithm based on Huffman coding. *J. Discret. Math. Sci. Cryptogr.* **2020**, *25*, 1–14. [CrossRef]