1

Introduction to Classical Cryptography

Vani Rajasekar[1]*, Premalatha J.[2], Rajesh Kumar Dhanaraj[3] and Oana Geman[4]

[1]Department of CSE, Kongu Engineering College, Tamil Nadu, India
[2]Department of IT, Kongu Engineering College, Tamil Nadu, India
[3]School of Computing Science and Engineering, Galgotias University, Greater Noida, India
[4]Department of Health and Human Development Computers, University of Suceava, Suceava, Romania

Abstract

In today's world, with societies of information that transmits a growing number of personal data via public channels, the protection of information is a global challenge. It is the technique used to authenticate users among two parties in the public environment, where there are unauthorized users and suspicious activities. Two types of process involved in classical cryptography are encryption and decryption, which are performed at the sender and receiver sides, respectively. Encryption is the method by which legible data or plain text is combined with key (additional information) and converted into an illegible data or cipher text. Decryption is the method by which cipher text is again transformed to plain text. Classical cryptography depends on mathematics and on the complexity of computing factorization in large numbers. The two major categories of classical cryptography are symmetric key cryptography and asymmetric key cryptography. In symmetric classical, both encryption and decryption are done using same key called private key [1]. In asymmetric classical, the sender uses public key to encrypt the message and the receiver uses private key to decrypt the message. The most widely used conventional cryptographic scheme is Data Encryption Standard (DES), Advanced Data Encryption Standard (AES), and Rivest-Shamir-Adleman (RSA). These algorithms have a primary advantage of flexibility and their protection, based on computational and validated security claims. The disadvantage relies on security of protocols against adversary.

**Corresponding author*: vanikecit@gmail.com

Rajesh Kumar Dhanaraj, Vani Rajasekar, SK Hafizul Islam, Balamurugan Balusamy and Ching-Hsien Hsu (eds.) Quantum Blockchain: An Emerging Cryptographic Paradigm, (1–30)

Message authentication code (MAC) plays an important role in classical cryptography. MAC value ensures the authenticity and data privacy of message, and it will detect any changes in the transformed messages. Secure Hash Algorithm, also known as SHA, is a family of cryptographic approaches that is used to keep the information secured [2]. SHA and MAC functions generally include bitwise operations, compression functions, and modular operations. All cryptographic approaches should fulfill the security features such as confidentiality, integrity, privacy, reliability, authentication, and non-repudiation. Cryptography tools are far more effective in the instances of signatures confirmation, user authentication, and to conduct other cryptography practices. The most extensively used classical cryptographic tools are security token, Java Cryptographic Libraries (JCA), Docker, and authentication keys. The major applications of cryptography implies banking, digital currencies, military communications, secure network communications, disk encryption, health care, education, software, and marketing.

***Keywords*:** Security, cryptography, authentication, privacy, key agreement

1.1 Introduction

The term "cryptography" is used to refer to the branch of science that sprang from historical secret writing. Secret writing is just one of many challenges for which cryptography has provided a solution. Cryptographic primitives relate to the answers to various cryptographic difficulties. The term "symmetric encryption" relates to a cipher in which the transmitter and receiver maintain a common secret key, allowing them to exchange a message in such a way that an adversary cannot decipher the message even if he witnesses it. In cybersecurity, a classical cipher is a form of encryption that has been employed in the past but has mostly fallen out of favors. Most classical ciphers, unlike modern cryptographic techniques, can be effectively calculated and decoded manually. However, with technological advances, they are usually fairly easy to break. The various classical ciphers are substitution cipher and transposition cipher.

1.2 Substitution Ciphers

A substitution cipher is a type of encryption in which plaintext elements are substituted with cipher text in a predetermined order using a key. The key units can be single letters, groups of letters, triplets of letters, combinations of the above, and so on. To retrieve the original message, the receiver uses the reverse substitution technique to decrypt the text. The plaintext

elements are reorganized in a different, generally very complex order in a transposition cipher, but the elements themselves remain unaffected [3]. In a substitution cipher, on the other hand, the plaintext elements are kept in the same order in the cipher text, but the values themselves are changed.

1.2.1 Caesar Cipher

Caesar cipher, also referred as the shift cipher, Caesar's coding, or Caesar shifting, is among the most basic and extensively used encryption algorithms. It is a substitution cipher where each letter in the plaintext is chosen to replace that is a certain number of places down the alphabet. The conversion can be visualized by aligning the alphabets. The cipher letter is the plain letter rotated by a certain range of positions left or right. For example, here, a Caesar cipher uses a three-place left rotation, which is identical to a three-place right shift.

Encryption:
The encryption can alternatively be described using modular arithmetic by converting the letters into integers using the A = 0, B = 1, …, Z = 25 technique. The mathematical formula for encrypting a letter x with a shift n is

$$E(x) = (x + n) \bmod 26 \tag{1.1}$$

Similarly, the decryption is

$$D(x) = (x - n) \bmod 26 \tag{1.2}$$

Plain	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Cipher	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Example:
Sample plain text: HELLO WORLD
Cipher Text: OLSSV AVYSK

In a cipher text–only environment, the Caesar cipher is easily broken. There are two scenarios to consider:

1. An attacker is aware that a simple substitution cipher has been employed.

2. An attacker recognizes the presence of a Caesar cipher but is unaware of the shifting value.

1.2.2 Polyalphabetic Cipher

Any cipher based on substitution that uses numerous substitution alphabets is known as a polyalphabetic cipher. Although it is a reduced particular case, the Vigenère cipher is possibly the best illustration of a polyalphabetic encryption [4].

1.2.2.1 Working of Polyalphabetic Cipher

- Select a keyword
- Construct the Vigenère table as shown in Figure 1.1.
- Write the key for plain text
- For each letter of plain text and a letter in key, look at the Vigenère table
- Trace down all the letters and finally write the cipher text

ROW																									
0	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
2	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
3	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
4	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
5	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
6	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
7	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
8	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
9	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
10	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
11	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
12	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
13	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
14	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
15	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
16	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
17	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
18	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
19	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
20	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
21	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
22	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
23	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
24	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
25	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Figure 1.1 Vigenère table.

Example:
Key: KEYKEYE
Plain text: TRYTHIS

Then, based on the steps described above and from the Vigenère table, it is identified that the cipher text as follows:

Cipher: DVWDLGW

1.2.2.2 Cracking of Cipher Text

- Look for repeated sequences of letters in the ciphertext; the longer the sequences, the stronger. Make a note of where they are by underlining or marking them in the some fashion [5, 6].
- Determine how many letters are between the first letters in the string and add one for each appearance of a repetitive string.
- Factor the number obtained in above calculation.
- Create a table of common components by repeating this method with each repeated string that has found. The length of the phrase used to encrypt the cipher text is perhaps the most common factor.
- Count the number of times each letter appears in the cipher text. There should be n separate frequency counts in the end.
- To determine exactly how much each letter was displaced, compare these counts to regular frequency distribution.
- Shift should be undoing and the message can be read.

1.2.3 Hill Cipher

Hill cipher is a linear algebra–based polygraph substitution cipher. A value modulo 26 is assigned to each letter. The basic scheme A = 0, B = 1, …, Z = 25 is frequently employed, although it is not a requirement of the encryption [7]. Encryption is done by multiplying plain text matrix with the key matrix as nXn. The decryption is done by inverting the key matrix and it should be multiplied with cipher text to produce the plain text.

Example: Encryption
Plain text: ACT
Key: GYBNQKURP

$$Plain\,text = \begin{bmatrix} 0 \\ 2 \\ 19 \end{bmatrix} \text{ and } Key = \begin{bmatrix} 6 & 24 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{bmatrix}$$

$$Cipher\,text = \begin{bmatrix} 6 & 24 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{bmatrix} \times \begin{bmatrix} 0 \\ 2 \\ 19 \end{bmatrix} \bmod 26$$

$$= \begin{bmatrix} 15 \\ 14 \\ 7 \end{bmatrix}$$

Cipher Text: POH

Example: Decryption
Cipher Text: POH
Find the inverse of key as follows:

$$Key^{-1} = \begin{bmatrix} 6 & 24 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{bmatrix}^{-1} \bmod 26 = \begin{bmatrix} 8 & 5 & 10 \\ 21 & 8 & 21 \\ 21 & 12 & 8 \end{bmatrix}$$

$$Plain\,text = \begin{bmatrix} 8 & 5 & 10 \\ 21 & 8 & 21 \\ 21 & 12 & 8 \end{bmatrix} \times \begin{bmatrix} 15 \\ 14 \\ 7 \end{bmatrix} \bmod 26$$

$$= \begin{bmatrix} 0 \\ 2 \\ 19 \end{bmatrix}$$

Plain Text: ACT

1.2.4 Playfair Cipher

The Playfair cipher was the very first digraph stream cipher to be used in practice. Unlike standard ciphers, Playfair cipher encrypts a pair of alphabets (digraphs) rather than a single alphabet. It starts by generating a 5*5 matrix key table. The matrix comprises alphabets that serve as the plaintext's encryption key. It is important to remember that no alphabet should be duplicated. Another thing to keep in mind is that there are 26 alphabets and only 25 blocks to fit a letter into. The message is encrypted digraph by digraph with the Playfair cipher. As a result, the Playfair cipher might be considered a digraph substitution cipher [8, 9].

1.2.4.1 Rules for Encrypting the Playfair Cipher

- To begin, divide the plaintext into digraphs (pair of two letters). Insert the letter Z to the start of the plaintext if it has an odd number of letters.
- If the letters in digraph appears twice then add X in between the letters
- To figure out the cipher text, make a 5*5 key matrix or key table and complete it with alphabet letters.
- Then, the condition to be followed is a) replace each letter of the digraph with the letters adjacent to their right if a pair of digraph appears in the same row; b) replace each letter of the digraph with the letters below them if a pair of digraph appears in the same row; c) in this scenario, select a 3*3 matrix from a 5*5 matrix and present the elements at the two opposite corner if digraph appears in a different row and column.

Example:
Plain Text: COMMUNICATION
Key: COMPUTER

- Split the plain text into digraphs as CO MX MU NI CA TI ON
- Construct 5*5 key matrix as follows

C	O	M	P	U
T	E	R	A	B
D	F	G	H	I
K	L	N	Q	S
V	W	X	Y	Z

- For the first digraph, CO, the cipher is OM; for the second digraph, MX, the cipher is RM; for the third digraph, MU, the cipher is PC; for the fourth digraph, NI, the cipher is SG; for the fifth digraph, CA, the cipher is PT; for the sixth digraph, TE, the cipher is ER.
- The final cipher text is OMRMPCSGPTER

1.3 Transposition Cipher

A transposition cipher is an encryption technique in which the positions of plaintext units often characters or groups of letters are shifted according to a regular pattern, resulting in the cipher text being a permutation of the plaintext. That is, the elements' order is altered that is the plaintext is reordered. To encrypt, a bijective function is applied to the locations of the letters, and to decrypt, an inversion function is applied [10].

1.3.1 Columnar Transposition

In a columnar transposition, the information is typed out in fixed-length rows and then called out column by column, with the columns picked in a random sequence. A keyword is commonly used to determine the width of the rows and the recombination of the columns. Any empty spaces in a standard columnar transposition cipher are completed with nulls; any empty spaces in an uneven columnar transposition cipher are left blank. Finally, the information is recited in columns according to the keyword's sequence.

Example:
Keyword: ZEBRAS
Order is: 6 3 2 4 1 5
Plain Text: WE ARE DISCOVERED FLEET AT ONCE

The columnar grid is as follows:
6 3 2 4 1 5
W E A R E D
I S C O V E
R E D F L E
E A T O N C
E Q K J E U

In regular case, after filling the (QKJEU) values, the cipher text is EVLNE ACDTK ESEAQ ROFOJ DEECU WIREE.

In the irregular case, the left-out column values are not completed by null.

The columnar grid is as follows:
6 3 2 4 1 5
W E A R E D
I S C O V E
R E D F L E
E A T O N C
E

The cipher text value is EVLNA CDTES EAROF ODEEC WIREE.

To decipher it, the receiver must divide the message length by the key length to determine the column lengths. Then, rewrite the statement in columns, reformatting the key phrase to reorder the columns.

1.3.2 Rail Fence Transposition

The rail fence encryption is a type of transposition cipher named after the technique in which it is encoded. The plaintext is placed vertically and diagonally on consecutive "rails" of an imagined fence in the rail fence cipher, then moved up to the bottom.

Example:
Plain text: HELLO WORLD
Rail Fence Format:

```
H     L         O         R         D
      E         L         W         L
```

Cipher Text: HLORD ELWL

1.3.3 Route Cipher

The plaintext is printed out in a grid of required dimension in a route cipher and then taken off in a sequence specified in the key.

Example:
Key: Clockwise spiral inward starting from top right
Dimension: 3
Plain Text: HELLO WORLD HOW ARE YOU
H L O D W E U
E O R H A Y X
L W L O R O Z
Cipher Text: UXZOROLWL EHLODWE YAHROE

1.3.4 Double Transposition

A simple columnar transposition could be exploited by predicting possible column lengths, putting the information in columns, and then hunting for anagrams. As a result, a double transposition was frequently utilized to strengthen it [11]. It is just a columnar transposition done twice. Both transpositions can be done with the same key or with two distinct keys.

Example:
Keyword: STRIPE
Permutation: 564231
6 3 2 4 1 5
W E A R E D
I S C O V E
R E D F L E
E A T O N C
E
Cipher Text: CAEEN SOIAE DRLEF WEDRE EVTOC

1.4 Symmetric Encryption Technique

Symmetric encryption is a type of cryptography that uses only one key, known as the secret key, to encrypt and decrypt the information. The key must be exchanged between the organizations communicating using symmetric encryption so that it may be utilized in the decryption process. This encryption method varies from asymmetric encryption, which encrypts and decrypts messages using a key pair, one public and one private. Data is changed to a format that can be read by anybody who does not have the unique key to decrypt it using symmetric encryption methods. Once the message has been delivered to the intended receiver who holds the key, the algorithms reverses its effects, returning the message to its original and comprehensible state. The secret key used by both the sender and receiver can be a particular login credentials or a randomized string of numbers

and letters created using a secure random number generator (RNG) [12, 13]. There are two different kind of symmetric algorithms such as block and stream ciphers.

- Block Cipher:
  By the use of a secret key, fixed sequences of bits are encrypted in electronic communications blocks. While the data is encrypted, the system stores it in memory while waiting for entire blocks.
- Stream Cipher:
  Rather than being stored in the system database, information is secure as it streams.

Although symmetric encryption is an outdated kind of encryption, it is quicker and more accurate than asymmetric encryption, which strains networks due to data capacity limitations and excessive CPU usage. Symmetric cryptography is frequently included in bulk encryption or encoding enormous amounts of information due to its greater efficiency and greater speed. In the case of a network, the secret key may be used to encrypt or decrypt data exclusively by the database. Reported plaintext attacks, chosen-plaintext attacks, differential cryptanalysis, and linear cryptanalysis have all been known to be vulnerable to symmetric cyphers in the past [14]. The functions for each round should be carefully constructed to limit the odds of a successful attack. In the case of a database, the secret key may be used to encrypt or decrypt data exclusively by the database.

1.4.1 Key Management

The maintenance of cryptographic keys in a crypto algorithm is referred to as key management. Handling with the creation, transmission, storage, usage, removal, and restoration of keys falls under this category. Cryptographic protocol development, key servers, user procedures, and other pertinent protocols are all covered. Key management is the process of keeping track of cryptographic keys in a crypto algorithm. This includes dealing with the production, transfer, storage, use, removal, and recovery of keys. The construction of cryptographic protocols, key servers, user processes, and other relevant protocols are all discussed. Various types of keys are used by various authentication systems, and some systems utilize more than one. Asymmetric or symmetric keys are examples of these. The keys used in a symmetric key algorithm are the same for both encoding and decoding a message. Keys should be chosen carefully, disseminated, and stored in a secure manner. In contrary, asymmetric keys also known as

public keys are two separate keys that are mathematically linked. They are usually used in tandem to communicate.

1.4.2 Key Generation

The process of producing keys in cybersecurity is known as key generation. To encode and decode whatever information is being encrypted or decrypted, a key is used. A key generator, often known as a keygen, is a device or program that generates keys. Prescriptive algorithms and public key algorithms are both used in modern cryptography systems. A single shared key is used in symmetric-key methods, and keeping information secret necessitates keeping this key secret as well. A public key and a private key are used in public key algorithms. Anyone can have access to the public key. A sender uses encryption with the receiver's public key, which can only be decrypted by the owner of the private key. Wireless techniques like TLS and SSH utilize a mixture of the two since public key algorithms are substantially lower than conventional algorithms. One party obtains the other's public key and encodes a short bit of data.

1.4.3 Key Exchange

Key exchange, also known as key establishment, is a contractual method of exchanging cryptographic keys among parties involved to enable the usage of a cryptographic algorithm. If the transmitter and recipient choose to send and receive encrypted communications, then they both need to be able to encrypt and decrypt messages. The key exchange challenge is exchanging whichever keys or other data is required to establish a confidential channel of communication that no one else can copy. To share information confidentially, parties involved must first interchange the secret key, which allows each party to encrypt and decrypt the data before sending or receiving them. The main issue with symmetrical cryptography, also known as single-key cryptography, is that it necessitates the transmission of a secret key via authorized couriers, diplomatic luggage, or any other safe communication route.

1.4.4 Data Encryption Standard

The Data Encryption Standard (DES) is a symmetric key technique for digital information encryption. Despite the fact that its small key length of 56 bits renders it unsecure for applications, it has had a significant impact on cryptography. Following the publishing of an NSA-approved encryption

standard, it was quickly adopted around the world and subjected to extensive scholarly investigation. Controversy emerged due to confidential design features: the symmetric key block cipher design's relatively short key length, and the NSA's involvement, creating concerns about a backdoor. The NSA constructed the S-boxes that sparked those suspicions to remove a backdoor they knew about. The NSA, on the other hand, made sure that the key size was dramatically lowered so that a brute force attack could be used to break the cipher. The algorithm was subjected to a lot of academic scrutiny throughout time, which led to the contemporary understanding of block ciphers and their cryptanalysis.

1.4.4.1 Structure of DES

DES is a method that generates a fixed-length sequence of plaintext bits and converts into another cipher text bit string of the same size using a series of difficult operations. The block size throughout the case of DES

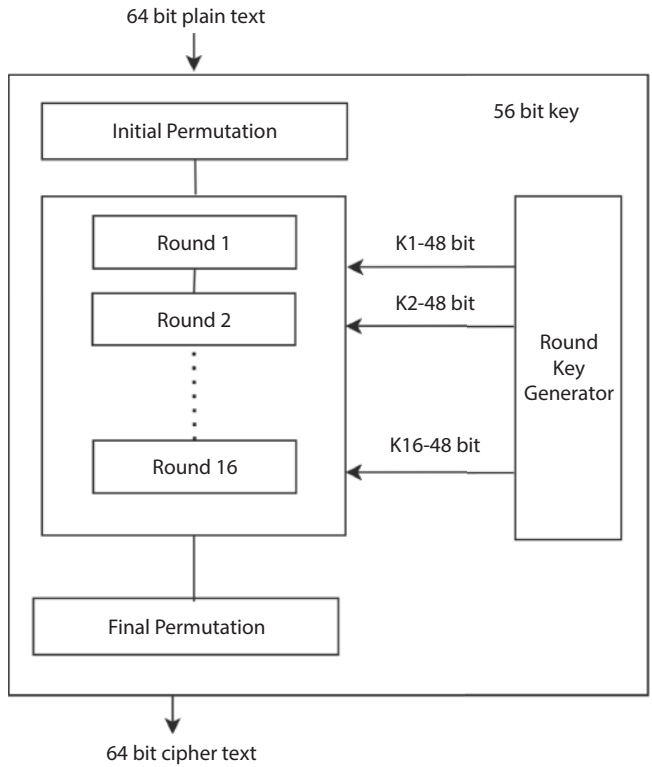

Figure 1.2 Structure of DES.

is 64 bits. DES also employs a key to modify the transformation, implying that decryption can only be accomplished by those who have access to the encrypting key. The overall structure of DES is specified in Figure 1.2. There are 16 rounds of operation, each with 16 identical phases. There are also inverses of the initial and final permutations, known as IP and FP. The block is broken into two 32-bit into halves treated simultaneously before the major rounds; this crossing is known as the Feistel scheme. Decryption and encryption are relatively comparable procedures according to the Feistel structure. The F-function encrypts a half-block and a portion of the key. The F-output function is then mixed with other half of the block before the following cycle, and the two strands exchanged [15]. The halves are exchanged after the final round; this is a property of the Feistel structure that makes encryption and decryption procedures identical.

1.4.4.2 Fiestel Function

The structure of Fiestel function is shown in Figure 1.3. The steps involved in the Fiestel function are shown in the following.

- Expansion:
  By replicating 50% of the bits, the 32-bit half-block is extended to 48 bits using the extension permutation, marked E in the image. The output is composed of eight 6-bit (86 = 48 bits) parts, each of which contains a copy of four general input bits as well as a copy of the directly adjacent bit of each of the input bits on each side.
- Mixing of key:
  An XOR function is used to merge the outcome with a subkey. To use the key schedule, sixteen 48-bit subkeys are produced from the main key, one in each round.
- Substitution
  The block is separated into eight 6-bit parts after adding in the subkey before even being processed by the S-boxes or substitution boxes. As shown in a non-linear adjustment available in the form of a lookup table, every one of the eight S-boxes substitutes its six input bits with 4 output bits. The S-boxes are at the heart of DES's security. The cipher would be linear and computationally breakable without them.
- Permutation
  Finally, the P-box rearranges the 32 outcomes from the S-boxes as shown in a predetermined permutation. This is structured such that the bits from each S-output box's in this round are

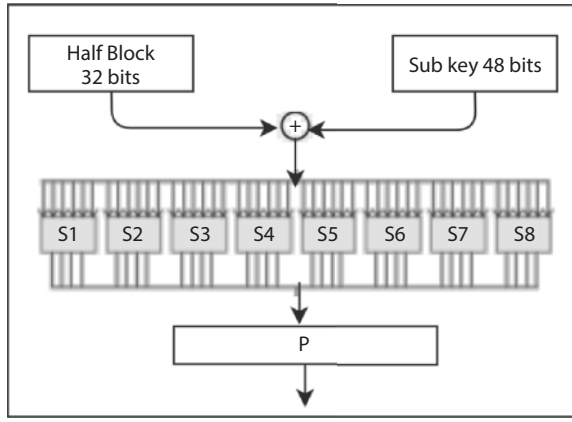

Figure 1.3 Structure of Fiestel function.

divided among four distinct S-boxes for the next round after recombination.

1.4.5 Advanced Encryption Standard

The Advanced Encryption Standard (AES), popularly known as Rijndael in its original form, is a standard for digital information encryption [16]. AES is based on a partial replacement network design idea and is effective in both hardware and software. AES is not using a Feistel network, unlike its predecessor DES. AES is a Rijndael variation with a 128-bit constant block size and a session key of 128, 192, or 256 bits. The state is a four-by-four column major-order matrix of bytes used by AES. The majority of AES calculations are performed in a finite field. The following is the number of rounds:

- For 128-bit keys, there are 10 rounds.
- For 192-bit keys, there are 12 rounds.
- For 256-bit keys, there are 14 rounds.

1.4.5.1 Operation in AES

The steps involved in the AES are described as follows:

- Key Expansion
  By using AES key schedule, round keys are produced from cypher keys. Each round of AES requires a distinct 128-bit rounds key block, with one additional.

- Initial Round Key
  Bitwise XOR is used to combine every byte of the value with a byte of the round key.
- Round operation
  1. Sub bytes
     Using an 8-bit substitution box, every byte in the state array is replaced with a sub byte in the sub bytes step. The cipher's non-linearity is provided by this process. The multiplicative inverse is often used to create the S-box. The S-box is built by merging the inverse function with an invertible affine transformation to resist assaults based on the simple mathematical features.
  2. Shift rows
     The shift rows step works on the state's rows, shifting the bytes in each row circularly by a given offset. The first row is unaffected for AES. The second row's bytes are displaced one to the left. Similarly, the third and fourth rows have two and three offsets, accordingly. As a result, each column of the shift rows step's output sequence is made up of bytes from each column of the input state. The purpose of this step is to prevent the columns from being encrypted separately, which would cause AES to devolve into four separate block ciphers.
  3. Mix column
     In the mix columns phase, an invertible transformation function is used to aggregate the four bytes of each state column. The mix columns function takes 4 bits as input and produces four bytes, with each input byte affecting each of the four output bytes. Mix columns in conjunction with shift rows enable cipher dispersion.
  4. Add round key
     The subkey is coupled with both the state in add round key step. Using Rijndael's key schedule, a subkey is generated from the main key for every round; each subkey is the same length as the state. The subkey is created by using bitwise XOR to combine each byte of the state with the appropriate byte of the subkey.
- Final Round
  The final round operation contains sub bytes, shift rows and add round key.

1.4.6 Applications of Symmetric Cipher

1. Financial applications such as banking transactions and smart card applications
2. Authentication applications
3. Hash code generation
4. Random number generation

1.4.7 Drawback of Symmetric Encryption

1.4.7.1 Key Exhaustion

Symmetric encryption has a flaw in which each use of a key "leaks" some information that could be exploited by an attacker to rebuild the key. The establishments of a key hierarchy to guarantee that master or key encryption secrets are not overused, as well as the proper rotation of keys that do encrypt large amounts of data, are two defenses against this behavior. Both of these techniques require competent key-management procedures to be traceable, as data might be compromised if an expired encryption key can still be recovered.

1.4.7.2 Key Management at Large Scale

When only some few keys, for example, tens to low hundreds are included in a scheme, the administration overhead is minimal and can be managed through manual, human work. With a large estate, however, keeping track of expiration dates and coordinating key rotation rapidly becomes impossible.

1.4.7.3 Attribution Data

With exception of asymmetric certificates, symmetric keys lack associated metadata such as an expiration date or an Access Rights to identify the key's intended usage in encryption but not in decryption.

1.5 Asymmetric Encryption Technique

Asymmetric key algorithms are comparable to symmetric key algorithms in that plaintext is coupled with a key, passed through an algorithm, and cipher text is returned. The main distinction is that the keys used for

encryption and decryption is distinct, resulting in the algorithm's asymmetry. A private key and a public key make up the key pair. Public key cryptography and digital signatures are the two most common applications of asymmetric key methods. Anyone can transmit an encrypted communication within a trusted network of users using public key encryption. The sender encrypts the communication with the receiver's public key, leaving only the receiver with his or her own shared secret key to decrypt it. In such a scheme, anyone can encrypt the message that use the designated receiver's public key, but only the recipient's private key can access this information. This allows server software to construct a cryptographic key for compatible symmetric key cryptography and then encrypt that freshly generated symmetric key using a client's openly provided public key. Robust authentication is also achievable with public key cryptography. A sender can construct a brief digital signature on a communication by combining it with a private key. Anyone having the sender's public key can compose a message with a claimed digital certificate, and if the signature is equal to the message, the message's origin is validated.

1.5.1 Rivest-Shamir-Adleman Encryption Algorithm

The Rivest-Shamir-Adleman (RSA) algorithm utilizes asymmetric cryptography, which means it employs both a public and a private key. A public key can be shared openly, but a private key is kept private and must not be communicated with anybody. The steps involved in the RSA are described as follows:

- Key Generation
  1. Select two large prime number p and q
  2. Compute $n = p \times q$
  3. Compute the totient function as $\phi(n) = p - 1 \times q - 1$
  4. Identify the integer e such that $1 < e < \phi(n)$
  5. Compute d such that $e.d = 1 \mathrm{mod} \phi(n)$
- Encryption
  1. Given a plain text message m, the cipher text c is calculated as $c = m^e \bmod n$
- Decryption
  1. Given the cipher text c, the plain text can be identified as $m = c^d \bmod n$

1.5.2 Elliptic Curve Cryptography

Elliptic curve cryptography (ECC) is a public key cryptography technique based on the analytical solution of elliptic curves over finite fields. In comparison to non-EC encryption, ECC allows for smaller keys. Key agreement, digital signatures, pseudo-random generators, and other jobs can all benefit from elliptic curves. By integrating the key agreement with a symmetric encryption algorithm, they could be used for encryption effectively. To utilize ECC, both parties will agree on all of the characteristics that compose the elliptic curve, i.e., the agreement's domain parameters. In the prime case, the data is characterized by p, and in the binary case, the pair of m and f. The elliptic coefficient is drawn by the characterizing equation's parameters a and b. Domain parameter creation is typically not performed by each individual because it requires computing the amount of instances on a curve, which is time-consuming and difficult to execute. As a result, various standard bodies produced elliptic curve domain parameters for a variety of field sizes. "labeled curves" or "standardized curves" are terms for domain parameters that can be referred by name or by the unique object identification defined in the standard standards. The structure of elliptic curve is shown in Figure 1.4.

1.5.2.1 Elliptic Curve

ECC is a comprehensive cryptography technology that is an alternative to RSA. It uses elliptic curve mathematics to generate security between key pairs for public key encryption. RSA uses prime numbers instead of elliptic curves to achieve a similar result, but ECC has recently gained prominence due to its reduced key size and ability to retain security. In opposition to RSA, ECC builds its public key cryptography techniques on the algebraic structure of elliptic curves over finite fields. As a result, ECC generates keys

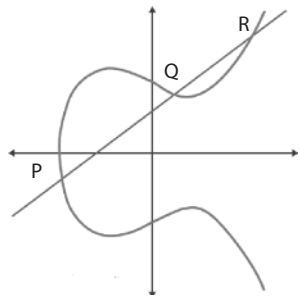

Figure 1.4 Elliptic curve.

that are computationally harder to break. As a result, ECC is regarded the next level of public key cryptography, and it is more secured than RSA. It also makes perfect sense to use ECC to keep both reliability and functionality at a high level. This is because ECC is becoming more widely used as websites aim for improved online security in client data while also improving mobile optimization. The demand for a concise guide to ECC grows as more sites use ECC to encrypt data. For modern ECC applications, an elliptic curve is a graphical method over a finite field composed of points obeying the formula:

$$y^2 = x^3 + ax + b$$

Whatever point on the curve can be duplicated over the x-axis in this ECC demonstration, and the curve will remain the same. Any non-vertical line that intersects the curve in three or fewer locations will be considered non-vertical.

1.5.2.2 Difference Between ECC and RSA

The size of RSA and ECC encryption keys differs significantly in terms of security output. In other terms, a 384-bit ECC key provides the same measure of protection as a 7,680-bit RSA key. The lengths of ECC and RSA keys do not have a linear relationship. That example, doubling the size of an RSA key does not result in doubling the size of an ECC key. This significant difference demonstrates that ECC key length and signing are far faster than RSA, as well as the fact that ECC utilizes less memory than RSA. Furthermore, unlike RSA, where both private and public keys are numbers, the private and public keys in ECC are not interchangeable equally. Conversely, the public key in ECC is a curve line, while the secret key is indeed an integer. ECC has shorter cipher texts, keys, and signature, as well as quicker key and signature production. It has a reasonably fast decryption and encryption performance. By computing signatures in two steps, ECC achieves lower latency than inverse. ECC has solid standards for authenticated key exchange, and the technology has a lot of backing.

1.5.2.3 Advantages and Security of ECC

The methods used in public key cryptography are simple to process in one direction but complicated to perform in the other. For example,

RSA makes use of the fact that combining prime numbers to generate a larger number is simple, but converting massive groups back to primes is significantly more complex. Elliptic curve encryption has a significant strength advantage, which translates to even more power for smaller, smart phones. Because factoring is easier and takes less energy than solving for an elliptic curve discrete logarithm, RSA's factoring authentication is more susceptible for two keys of same length. ECC has various flaws, notably side-channel attacks and twist security attacks. These types try to compromise the ECC's secret key security. Exploitable vulnerabilities are common as a result of side-channel attacks such as differential power assaults, fault analysis, normal power attacks, and simple temporal attacks. All sorts of side-channel attacks have simple responses. The twist security vulnerability, also known as a fault attack, is another sort of elliptic curve attack. Invalid-curve attacks and small-subgroup attacks are examples of such assaults, and they can expose the victim's secret key. Twist security threats are often averted by carefully validating parameters and selecting curves.

1.5.3 Hyperelliptic Curve Cryptography

In the same way that the Jacobian of an elliptic curve is an Abelian group with which to do computation, hyperelliptic curve cryptography (HECC) is analogous to ECC in that the Jacobian of a hyperelliptic curve is an Abelian group in which to do computation. An hyperelliptic curve over g is given by

$$C{:}y^2 + h(x)y = f(x)$$

The Jacobian of C, written J(C), is a division group; therefore, its elements are similarity classes of divisors of degree 0 under the connection of linear equivalence, rather than points. This is consistent with the elliptic curve situation, because the Jacobian of an elliptic curve can be proven to be isomorphic to the set of points on the elliptic curve. Neal Koblitz pioneered the use of hyperelliptic curves in encryption in 1989. Despite being released only three years after ECC, few cryptosystems use hyperelliptic curves since the arithmetic application is not as effective as with elliptic curves or factoring. Like an Abelian group, the Jacobian on a hyperelliptic curve could be used to solve the discrete logarithm issue (DLP). This allows for the usage of Jacobians of relatively low order, making the service more effective. However, if the hyperelliptic curve is chosen incorrectly, then the DLP becomes trivial to solve. The DLP in the Jacobian of

hyperelliptic curves could be attacked using any generalized attack on the discrete logarithm in finite Abelian groups, such as the Pohlig–Hellman technique and Pollard's rho approach. Another approach that can be used to resolve DLP in some cases is the indexing calculus algorithm. There is an index calculus assault on DLP for Jacobians of hyperelliptic curves. The approach will be more effective than Pollard's rho if the genus of the curve grows too large. In light of numerous DLP attacks, it is feasible to compile a list of hyperelliptic curve characteristics that should be ignored. However, as shown, DLP in this addition group is straightforward to solve. These curves, referred to as anomalous curves, are likewise not to be employed in DLP.

1.6 Digital Signatures

A digital signature is a computational method for verifying the integrity and validity of a message, program, or digital document. It is the virtual counterpart of a written document or a stamped seal, but it comes with a lot more security built in. The purpose of a digital signature is to prevent manipulation and deception in electronic communications. Electronic papers, transactions, and digital messages can all benefit from digital signatures as proof of origin, identity, and position. They can also be used to affirm informed permission by signers. Digital signatures are a ubiquitous feature of most cryptographic protocol suites, and they are often used for cloud hosting, financial transactions, contract software solutions, and other situations where forgery or tampering must be avoided. Digital signatures are a ubiquitous feature of most cryptographic protocol suites, and they are often used for cloud hosting, financial transactions, contract software solutions, and other situations where forgery or tampering must be avoided. Asymmetric cryptography is used in electronic certificates. They give a layer of validation and security to communications sent via an insecure channel in several cases: A digital signature, when properly implemented, gives the recipient reason to assume the communication was transmitted by the stated sender. In many ways, digital signatures are comparable to conventional handwritten signatures.

1.6.1 Working of Digital Signature

Public key cryptography, often known as cryptographic algorithms, is used to create digital signatures. Two keys are produced using a public key algorithm like RSA resulting in a mathematically connected pair of

keys, one secret and one public. The two or more mutually authenticating encryption algorithms of public key cryptography are used to create digital signatures. The person who makes the digital signature encodes signature-related data with a secret key, which can only be decrypted with the signer's public key. If the receiver cannot open the documents using the signer's public key, then there is an issue with the signature or the message. Digital signatures are verified in this way. All parties must trust that the person who creates the signature has maintained the private key secret in order for digital signature technology to work.

Typically, a digital signature approach consists of three methodologies:

- A key generation technique that chooses a private key from a collection of potential private key at random. The technique generates both a private key and a public key.
- Given a text and a private key, this signing method generates a signature.
- A signature verification algorithm that tries to determine the message's claim to legitimacy based on the message, public key, and signature.

1.6.2 Creation of Digital Signature

Signature software, such as an email application, is used to produce a digital signature by providing a one-way hash of the digital information to be signed. An algorithm generates a fixed-length bunch of numbers and integers called a hash. The hash is then encrypted using the private key of the digital signature originator. The digital signature is comprised of the encoded hash, as well as other characteristics such as the hash function. Because a hash function can turn any input into a specified result, which is usually significantly shorter, it is preferable to encode the hash rather than the full message or document. Hashing is much quicker than signing; therefore, this saves the time. A hash's value is distinct to the data it hashes. Any modification in the data, even a specific character modification, will result in a new value. This property allows others to decode the hash using the signer's public key to verify the information's authenticity. It shows that the data has not altered since it was verified if the decrypted hash matches a subsequent computed hash of the same data. If the two hashes does not equal, then the data was either changed with and is now vulnerable, or the signature was made with an encryption key that does not match the decryption key supplied by the signer, resulting in an authentication problem. The process of creating digital signature is denoted in Figure 1.5.

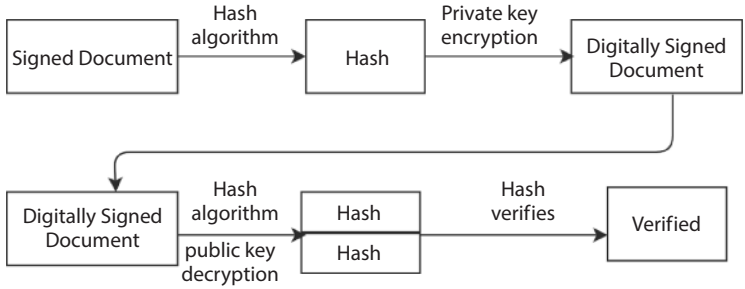

Figure 1.5 Process of creating digital signature.

A digital signature could be used with any type of message, if encrypted or not, to ensure that the sender's identity is verified and that the message was delivered intact. Because the digital signature is distinctive to both the document and the signer, it connects them altogether; it is hard for the signer to dispute finally signed something. Non-repudiation is the term for this characteristic. Digital signatures and message authentication are not the same thing. A digital certificate is a digital document that includes the granting CA's digital signature. It connects a public key to an individual or entity's identification which can be used to validate that a person belongs to that individual or persons. Digital signatures and digital certificates are supported by most web-based email applications, making it simple to sign any outgoing emails and authenticate digitally signed receiving communications. Digital signatures are also widely employed to give assurance of validity, integrity of data, and non-repudiation of online interactions and experiences.

1.6.3 Message Authentication Code

A message authentication code (MAC), also referred as a tag in cryptography, is a brief piece of knowledge used to authenticate a message—that is, to ensure that it originated from the specified sender and has not been modified. By enabling verifiers to identify any modifications to the message content, the MAC value safeguards both the data integrity and the validity of the communication. The MAC algorithm is a communication authentication symmetric key encryption mechanism. The transmitter and receiver exchange a symmetric key K to initiate the MAC process. A MAC is an encoded checksum computed on the underlying message being sent along with message. The sender enters the information and the secret key K into a publicly available MAC algorithm and generates a MAC value. The MAC

function compressing an arbitrary lengthy input into a fixed size output, similarly to the hash function. The key difference among hash and MAC is that MAC compresses using a shared secret. The message is forwarded together with the MAC by the sender. We will assume the information is sent in the clear because we are only interested in message origin verification, not message secrecy. If message confidentiality is essential, then the message must be encrypted. The receiver inserts the received message and the sharing secret key K into the MAC technique and recomputed it after receiving the message and the MAC. The receiver now compares the newly computed MAC to the MAC received from the sender to ensure that they are equal. If they are identical, then the receiver receives the message and is confident that it was sent by the intended sender. If the calculated MAC differs from the MAC given by the sender, then the recipient is unable to tell if the message has been tampered with or whether the origin has been forged. In the end, a recipient can reasonably conclude that the message is not real.

1.6.3.1 Limitation on MAC

- Establishing shared secret: It can authenticate messages between pre-determined legitimate individuals who share a key. Prior to using MAC, a shared secret must be established.
- Lack of non-repudiation: The guarantee that a message originator cannot disavow any previously communicated messages, agreements, or acts is known as non-repudiation. A non-repudiation service is not provided by the MAC approach. MACs cannot give verification that a communication was sent by the sender if the transmitter and recipient have a disagreement about message origination.

1.6.3.2 One Time MAC

As soon as the key is only used once, global hashing and, in particular, bilateral separate hash functions provide one reliable message authentication. This can be thought of as an authenticating one-time pad. The random key, key = (a, b), is the shortest pairwise independent cryptographic algorithm, and the MAC tag for a message m is calculated as tag = (am + b) mod p, where p is prime. For k-ways autonomous hashing functions, k-independent hashing functions give a secure MAC if the key is utilized less than k times. In the context of quantum cryptography, hash function and data access control have also been proposed. In contradiction to other

cryptographic tasks, such as digital certificates, quantum resources have been demonstrated to have no advantage over absolutely secure one-time conventional MACs over a rather extensive class of quantum MACs.

1.6.4 Secure Hash Algorithm

Secure Hash Algorithm (SHA) is a group of cryptographic techniques that are used to keep data safe. It operates by converting data with a hash function, which is a bitwise operations, modular additions, and compression function–based method. The hash function then generates a fixed-length string that bears no resemblance to the original. These techniques are one-way functions, which mean that it is nearly impossible to change them back into its original data once they have been changed into their feature vectors. SHA-1, SHA-2, and SHA-3 are three algorithms of importance, each of which was built with increasingly prominent encryption in reaction to hacker attempts. Because of its widely known flaws, SHA-0 is currently considered obsolete [17]. A common application of SHA is to encrypt passwords, as the server side just needs to maintain track of a physical requirements hash value, rather than the actual password. If the database is hacked, an attacker will only be able to see the hash functions, not the real credentials. Furthermore, SHAs display the avalanche effect, in which a small change in the encrypted letters causes a large change in the output; or, conversely, vastly dissimilar strings yield comparable hash values. As a result of this consequence, hash values do not provide any information about the input string, such as its initial size.

1.6.4.1 Characteristics of SHA

Pre-image resistant, second pre-image resistance, and collision resistance are three key safety features used by cryptographic algorithms to keep the information secure. The supply of pre-image resistance particularly makes it difficult and time-consuming for an attacker to find an original comment given the hash value and is the cornerstone of cryptography algorithms. The structure of one-way functions, which is a crucial component of SHA, provides this security. To fend off brute force assault from strong machines, pre-image resistance is required [18]. Second pre-image resistance is a security measure offered by SHA when a communication is known but it is difficult to locate another message that hashes algorithms to the same value. Collision resistance is the final safety feature, which is given by methods that make it much more difficult for an adversary to locate two very different communications that hashes to the same hash value. In addition

to provide this feature, there must be a comparable number of potential inputs as random variables, as more inputs than outputs will inevitably result in potential collisions according to the pigeonhole principle.

1.6.4.2 Applications of SHA

TLS and SSL, PGP, SSH, S/MIME, and IPsec are just a few of the security applications and devices that use SHA-1. MD5 and SHA-1 are both developed from MD4 and can be used in those applications. The hash techniques SHA-1 and SHA-2 are legally required for use in certain defense department applications, including use inside other cryptographic methodologies, in order to secure sensitive confidential information. FIPS PUB 180-1 also recommended private and commercial enterprises to adopt and use SHA-1.

1.6.5 Advantages and Disadvantages of Digital Signature

If some crucial conditions are handled both before and during the signing and verification procedure, then digital certificates are legally enforceable [19, 20]. To begin, signers must establish their identification before the document may be considered legally enforceable.

1.6.5.1 Advantages of Digital Signature

- A digital signature is more secure than an electronic system because it uses unique identification keys to construct and validate the signature.
- The time-consuming process of creating a digital signature offers a higher level of security, which is suitable for sensitive data.
- Digital signatures are widely accepted around the world and are legally enforceable in the majority of countries.
- Unauthorized users cannot change digitally signed papers or information.

1.6.5.2 Disadvantages of Digital Signature

- The technology utilized to establish a digital signature will have a big impact on it. If technology develops at its current rate, digital signatures must evolve at the same rate or risk losing their effectiveness.

- Digital signatures need the purchase of digital certificates, which can be rather costly.
- Users must also invest in verification software.

1.6.6 Conclusion

This chapter represents the various classical cryptographic techniques. Classical cryptography depends on mathematics and on the complexity of computing factorization in large numbers. The two major categories of classical cryptography are symmetric key cryptography and asymmetric key cryptography. In symmetric classical, both encryption and decryption are done using same key called private key. SHA is a group of cryptographic techniques that are used to keep data safe. It operates by converting data with a hash function, which is a bitwise operations, modular additions, and compression function–based method.

References

1. Yin, J., Li, Y.-H., Liao, S.-K., Yang, M., Cao, Y., Zhang, L., Ren, J.-G., *et al.*, Entanglement-based secure quantum cryptography over 1,120 kilometres. *Nature*, 582, 7813, 501–505, 2020.
2. Arnon-Friedman, R., Dupuis, F., Fawzi, O., Renner, R., Vidick, T., Practical device-independent quantum cryptography via entropy accumulation. *Nat. Commun.*, 9, 1, 1–11, 2018.
3. Shenoy-Hejamadi, A., Pathak, A., Radhakrishna, S., Quantum cryptography: key distribution and beyond. *Quanta*, 6, 1, 1–47, 2017.
4. Wang, L.-J., Zhang, K.-Y., Wang, J.-Y., Cheng, J., Yang, Y.-H., Tang, S.-B., Yan, D., *et al.*, Experimental authentication of quantum key distribution with post-quantum cryptography. *NPJ Quantum Inf.*, 7, 1, 1–7, 2021.
5. Rajasekar, V., Premalatha, J., Sathya, K., Multi-factor signcryption scheme for secure authentication using hyper elliptic curve cryptography and bio-hash function. *B. Pol. Acad. Sci. Tech.*, 68, 923–935, 2020.
6. Larocque, H., Gagnon-Bischoff, J., Mortimer, D., Zhang, Y., Bouchard, F., Upham, J., Grillo, V., Boyd, R.W., Karimi, E., Generalized optical angular momentum sorter and its application to high-dimensional quantum cryptography. *Opt. Express*, 25, 17, 19832–19843, 2017.
7. Shang, T., Tang, Y., Chen, R., Liu, J., Full quantum one-way function for quantum cryptography. *Quantum Eng.*, 2, 1, e32, 2020.
8. Rajasekar, V., Jayapaul, P., Krishnamoorthi, S., Cryptanalysis and enhancement of multi factor remote user authentication scheme based on signcryption. *Adv. Math. Commun.*, 1–19, 2020. doi:10.3934/amc.2020103

9. Kumar, D.R., Krishna, T.A., Wahi, A., Health Monitoring Framework for in Time Recognition of Pulmonary Embolism Using Internet of Things. *J. Comput. Theor. Nanosci.*, 15, 5, 1598–1602, 2018.
10. Buchmann, J., Braun, J., Demirel, D., Geihs, M., Quantum cryptography: a view from classical cryptography. *Quantum Sci. Technol.*, 2, 2, 020502, 2017.
11. Rajasekar, V., Premalatha, J., Sathya, K., Cancelable Iris template for secure authentication based on random projection and double random phase encoding. *Peer Peer Netw. Appl.*, 14, 2, 747–762, 2021.
12. Moizuddin, M., Winston, J., Qayyum, M., A comprehensive survey: quantum cryptography, in: *2017 2nd International Conference on Anti-Cyber Crimes (ICACC)*, IEEE, pp. 98–102, 2017.
13. Velliangiri, S., Manoharn, R., Ramachandran, S., Rajasekar, V.R., Blockchain Based Privacy Preserving Framework for Emerging 6G Wireless Communications. *IEEE Trans. Industr. Inform.*, 2021. doi: 10.1109/TII.2021.3107556.
14. Lakshmi, P.S. and Murali, G., Comparison of classical and quantum cryptography using QKD simulator, in: *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, IEEE, pp. 3543–3547, 2017.
15. Billewar, S.R., Londhe, G.V., Ghane, S.B., Quantum cryptography: Basic principles and methodology, in: *Limitations and Future Applications of Quantum Cryptography*, pp. 1–20, IGI Global, Mumbai, 2021.
16. Rajasekar, V., Premalatha, J., Sathya, K., Saračević, M., Secure remote user authentication scheme on health care, IoT and cloud applications: A multilayer systematic survey. *Acta Polytech. Hung.*, 18, 3, 87–106, 2021.
17. Dhivya, M.N. and Banupriya, M.S., Network Security with Cryptography and Steganography. *Int. J. Eng. Res. Technol.*, 8, 3, 1–4, 2020.
18. Krishnasamy, L., Dhanaraj, R.K., Ganesh Gopal, D., Reddy Gadekallu, T., Aboudaif, M.K., Abouel Nasr, E., A Heuristic Angular Clustering Framework for Secured Statistical Data Aggregation in Sensor Networks. *Sensors*, 20, 17, 4937, 2020.
19. Keserwani, P.K. and Govil, M.C., A Hybrid Symmetric Key Cryptography Method to Provide Secure Data Transmission, in: *International Conference on Machine Learning, Image Processing, Network Security and Data Sciences*, Springer, Singapore, pp. 461–474, 2020, July.
20. Gupta, M., Gupta, M., Deshmukh, M., Single secret image sharing scheme using neural cryptography. *Multimed. Tools Appl.*, 79, 1–22, 2020.