



ABSTRACT

Cryptography is the process of encrypting messages and other data in order to transmit them in a form that can only be accessed by the intended recipients. It was initially applied to written messages. With the introduction of modern computers, cryptography became an important tool for securing many types of digital data. Day by day as the modernization is getting stronger therefore the encryption is becoming weaker.

So, we have stepped up a little study on strengthening SSS related cryptography. The humble motive of our study is to express SSS using Public key. Threshold cryptography enables the construction of reliable and robust key management system which can reconstruct the key even in the destroy of some shares and on the contrary the key cannot be reconstructed unless a predefined set of shares are been accumulated. The earlier schemes available in literature lead to high computational complexity during both sharing and reconstructing. Here we are suggesting a scheme which employs simple graphical masking method, done by simple ANDing for share generation and reconstruction can be done by simple XORing the qualified set of shares. We assume that the user will be known to SSS, DES, ANDing, XORing operations. We are encrypting the generated share by using Public Key Cryptography. Hence, we can increase the security of this encryption technique. Nevertheless, it confirms authenticity, confidentiality, non-repudiation and integrity as well.

What is Cryptography?

- Cryptography is the art and science of making a cryptosystem that is capable of providing information security.

Need of cryptography:

- Cryptography is used to secure transactions and communications, safeguard personal identifiable information and other confidential data, authenticate identify, prevent document tampering, and establish trust between servers.






Cryptography is the science of secrets. Literally meaning 'hidden writing,' cryptography is a method of hiding and protecting information by using a code, or cipher, only decipherable by its intended recipient. Today, modern cryptography is essential to the secure Internet, corporate cybersecurity, and blockchain technology. However, the earliest use of ciphers dates back to around 100 BC.



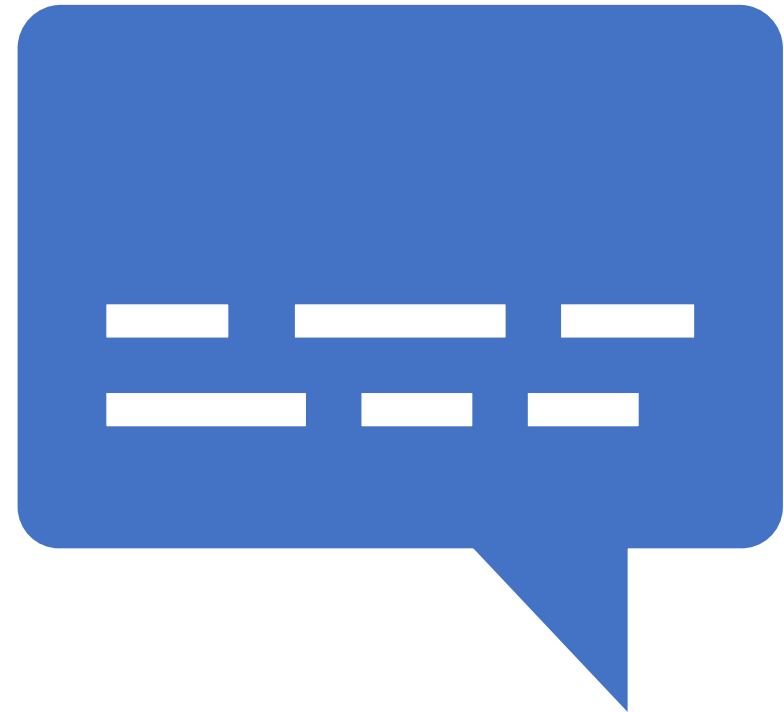
Historical Cryptography

Caesar Box

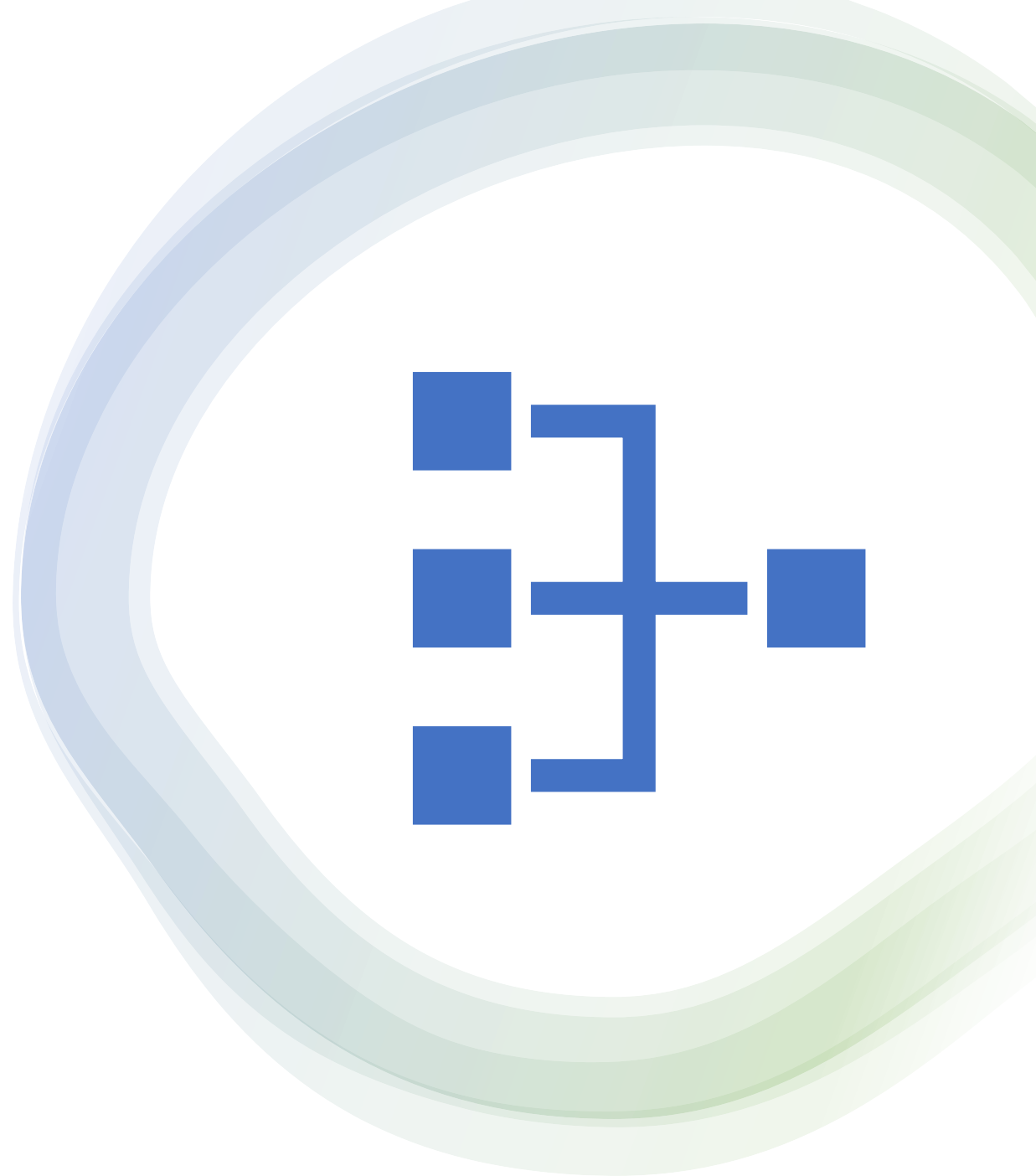
The "Caesar Box," or "Caesar Cipher," is one of the earliest known ciphers. Developed around 100 BC, it was used by Julius Caesar to send secret messages to his generals in the field. In the event that one of his messages got intercepted, his opponent could not read them. This obviously gave him a great strategic advantage. So, what was the code?



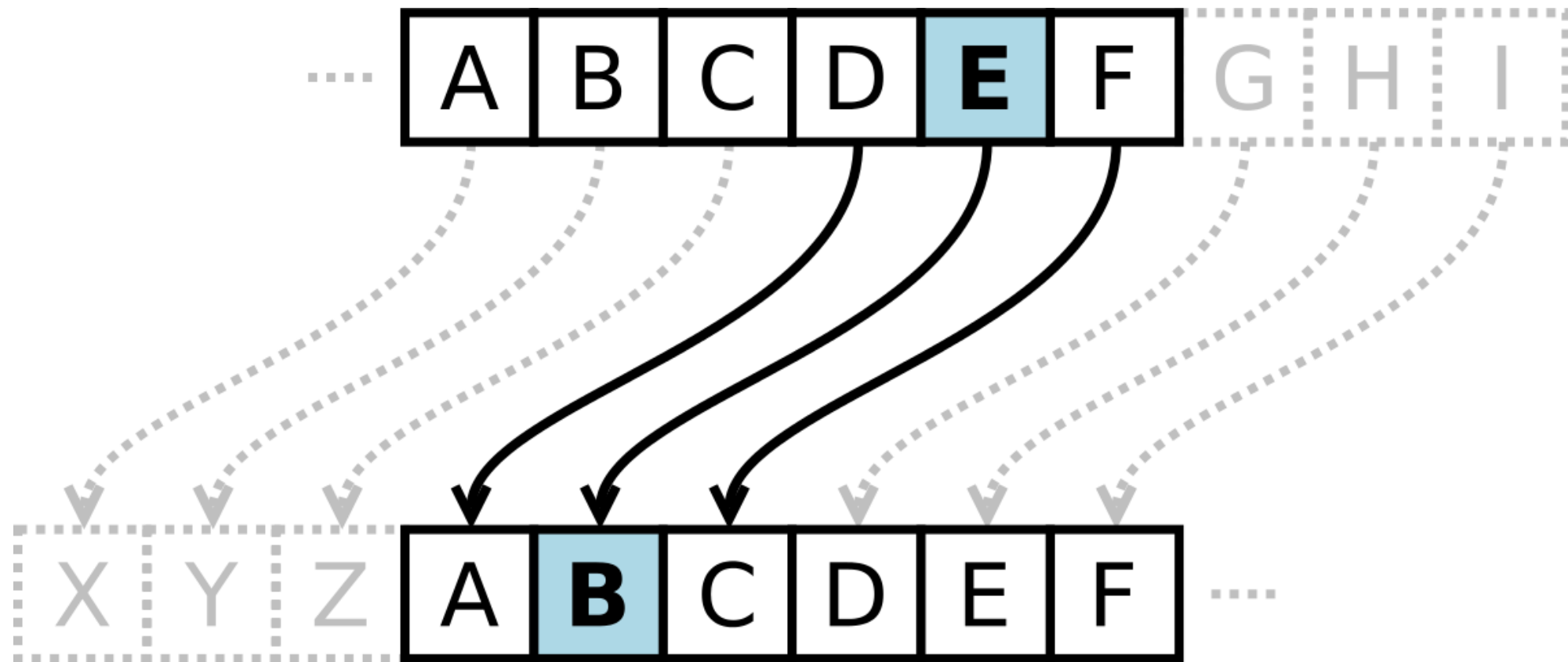
Caesar shifted each letter of his message three letters to the right to produce what could be called the ciphertext. The ciphertext is what the enemy would see instead of the true message. So, for example, if Caesar's messages were written in the English alphabet, each letter "A" in the message would become a "D," the "B's" would become "E's," and the "X's" become "A's." This type of cipher is appropriately called a "shift cipher."



Caesar's magic number was three, however, a modern day use of the Caesar Cipher is a code called "ROT13." ROT13, short for "rotate by 13 places," shifts each letter of the English alphabet 13 spaces. It is often used in online forums to hide information such as movie and tv show spoilers, solutions to puzzles or games, or offensive material. The code is easily crackable, however, it hides the information from the quick glance.



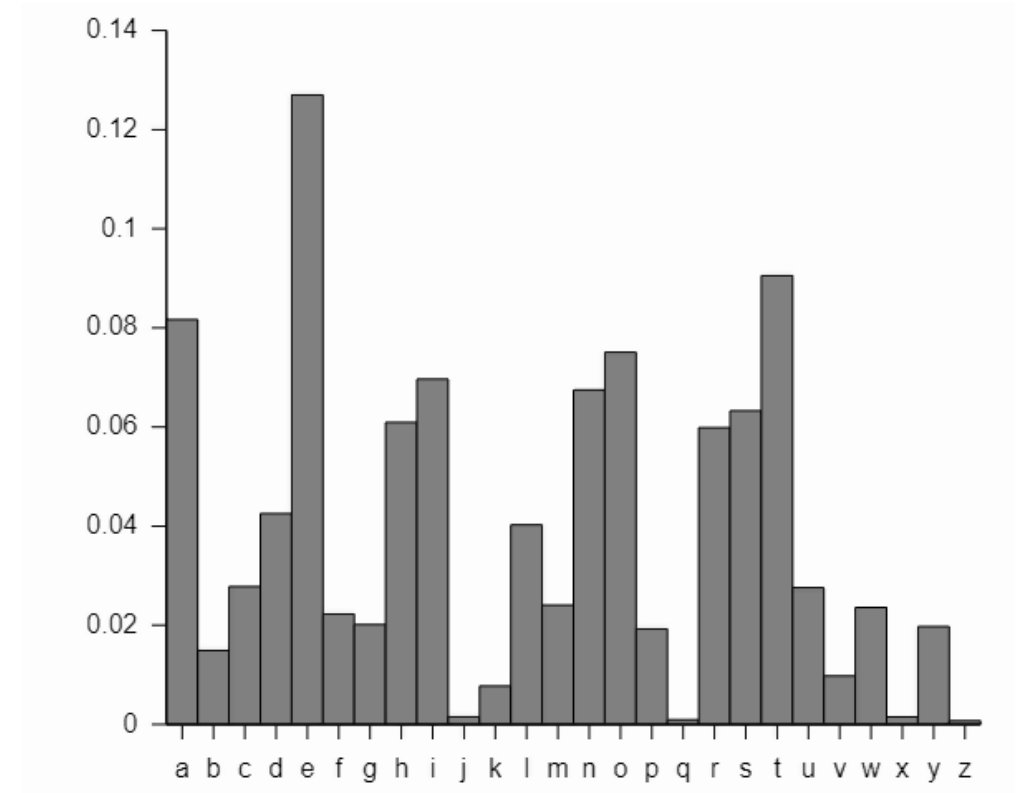
CAESAR CIPHER



Cryptanalysis: Breaking a Caesar Box

The hardest part of breaking a Caesar Box is figuring out the language of the message that it encodes. Once the code cracker figures this out, two scenarios are considered. Either the “attacker” utilizes a technique such as frequency analysis, or they use what is referred to as a brute force attack.

In the first instance, the attacker knows that certain letters are used more frequently than others. For example, A,E,O, and T are the most commonly used letters, while Q, X, and Z are the least. The relative frequencies of each letter in the English language are shown in the graph below.



If using a simple substitution cipher such as frequency analysis does not crack the code, an attacker could perform a brute force attack. This kind of attack entails testing each possible shift on a small snippet of the message. So, if the message is written in English, this would require a maximum of 26 tests since there are 26 letters in the English alphabet. While this is not a particularly sophisticated attack, it is effective.



The Vigenère Cipher

The Vigenère Cipher, created in the 16th century, uses an element not found in a Caesar Cipher: a secret key. The creator of the code picks any word or combination of letters at random to be the key, for example, "DOG." The keyword chosen will then be matched to the plaintext message that you want to encrypt, for example, "ATTACK." You can see that the keyword "dog" is shorter than the word "attack" by three letters. In this case, repeat your key until it matches the number of letters in your plaintext message. In this case, you would then have "DOGD OG."

Now, you will be able to create the ciphertext. To do this, you will need to use the chart below.

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

The columns are the letters of the secret key, while the rows are the letters of the plaintext message. So for our example, the first letter of our key is “D,” while the first letter of our plaintext word is “A.” So, find where they intersect on the chart, and you will find the first letter of our ciphertext, which is “D.” Next, the second letters of our key and plaintext words are “O” and “T” respectively. They intersect at “H.” You would continue this until you complete all six letters.

Plaintext Message: ATTACK

Key: DOGDOG

Ciphertext: DHZDQQ

Cryptanalysis: Cracking a Vigenère Cipher

Because of the use of a key, the Vigenère Cipher cannot initially be cracked by using a simple frequency analysis like you could do with a Caesar Cipher. Though, the main weakness of a Vigenère Cipher is the repeating of the key. So, in our example, “dog” was repeated twice in order to match the number of letters in the word “attack.” If an attacker guesses the key’s length, it becomes much easier to crack. The ciphertext is then treated like a series of small Caesar Ciphers, and a method such as frequency analysis could then be performed to crack the code.

But how can an attacker guess the length of the key? There are actually two methods: the Kasiski examination and the Friedman test. If the attacker notices that there are repeated segments of text in the ciphertext, a Kasiski examination would be effective in cracking the code. The attacker would count the distance of letters between repeated text to get a good idea of how long the key is. The Friedman test takes an algebraic approach utilizing a formula to measure the unevenness of the cipher letter frequencies to break the cipher. The longer the text, the more accurate this technique is

20th Century Cryptography

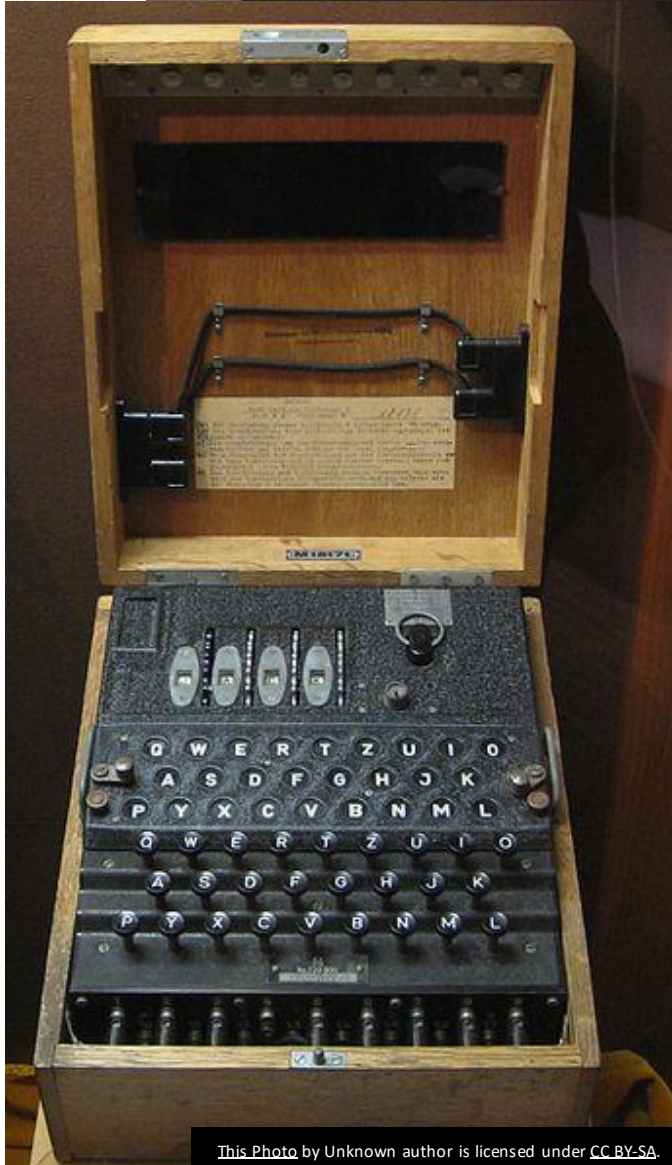
It wasn't until the 20th century that cryptography came into widespread use. Notable encryption milestones from the 20th century include the development and use of the Enigma machine, the creation of the US Data Encryption Standard (DES), and the development of asymmetric encryption.



The Enigma Machine

The Enigma machine is arguably the most famous encryption device to date. It was developed by Arthur Scherbius in 1918 at the end of WWI. The Enigma uses a rotor mechanism and special wiring to scramble the alphabet to easily and quickly create ciphertext from plaintext or vice versa.

When in use, one person types the message on the keyboard, while a second person writes down which letters light up on the light board directly above the keyboard. For example, you may type an "O" on the keyboard, but the "E" may light up on the light board. However, if you type "O" a second time, perhaps a "T" might light up. If you are typing the plaintext message on the keyboard, the letters from the light board would be the ciphertext or vice versa.



This Photo by Unknown author is licensed under [CC BY-SA](#)

But how does it work? The rotor mechanism changes the electrical connections between the keys and the lights with each keypress. To remain secure, the settings on the machine were changed daily based on secret key lists distributed in advance. The receiving station had to know and use the exact settings employed by the transmitting station to successfully decrypt a message. Decryption was accomplished via the same process.

The Enigma machines were famously used by the Germans in WWII to encrypt radio communications, but versions were also employed by the Japanese and Italians as well.

Cryptanalysis: Cracking the German Code

Breaking the encryption of the Enigma machine was a multistage process. The model in use by the Germans had cryptographic vulnerabilities which ultimately allowed a Polish cryptanalyst, Marian Rejewski, to crack the encryption keys in use. However, he did not have knowledge of the internal wiring of the machines, so he was not able to use this knowledge to break the codes. This information was actually obtained by a French spy, Hans-Thilo Schmidt, along with all of the codes used for encryption during September and October of 1932.

Using the newly obtained ciphertexts from this period, the Polish were able to build a working Enigma machine and decrypt German communications in three short months. This began a "cat and mouse game" between the Germans and the Polish until 1938, when German improvements made decryption too resource-intensive for the Polish to maintain.

In July of the following year, the Polish disclosed their Enigma decryption efforts to French and British military intelligence. This partnership allowed the British to develop their Enigma decryption efforts at Bletchley Park and continue decryption of German communications for the rest of the war.

Lucifer: The Data Encryption Standard

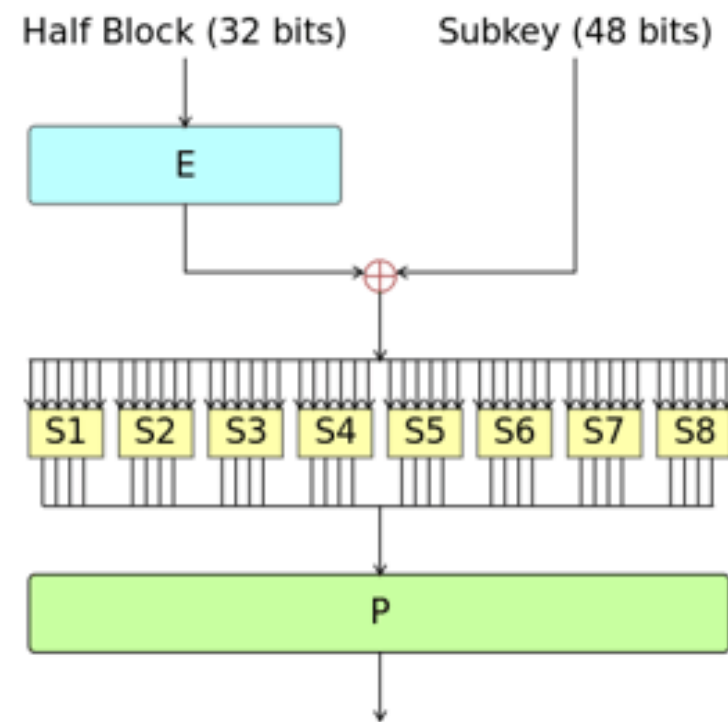
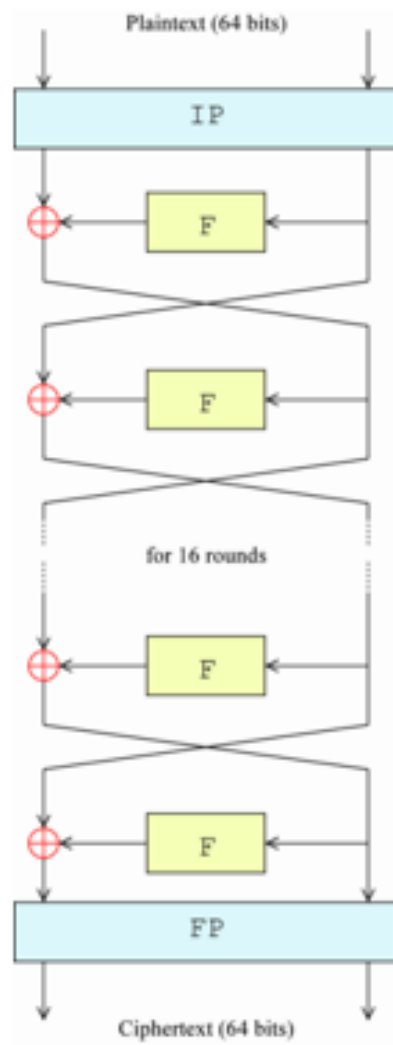
Before the early 1970's, the use of cryptography was primarily restricted to military use.

However, in 1971, the first encryption algorithms for private-use were published. By this time, companies had seen the effectiveness of encryption and were interested in applying it to the protection of their own intellectual property.

The first set of civilian encryption algorithms were developed by IBM under the name "Lucifer." Several different versions were created by IBM, one of which they submitted to the US National Bureau of Standards in hopes of getting it accepted as the national Data Encryption Standard (DES). The version they submitted was initially vulnerable to "differential cryptanalysis," but after some modifications done by the National Security Agency (NSA), IBM's Lucifer was indeed accepted as the Data Encryption Standard

The accepted version was a "Feistel network," which is a multi-round encryption algorithm whose structure is shown in the images below. The image on the right shows the operations that would be performed in each of the left image's boxes labeled "F" (for Feistel function).

So, the message undergoes a permutation at the beginning of the encryption process and at the end of the encryption process (boxes "IP" and "FP"), as well as during each round of the Feistel function (the green box labeled "P"). Getting technical, the "E" box in the right image represents an expansion of the half-block from 32- to 48-bits, while each of the numbered "S" boxes represents a substitution table that reduces 6 input bits to 4 output bits. The cipher also has a key schedule, which generates 16 48-bit round keys from the original 56-bit secret key.



Brute-Force Attack on the Data Encryption Standard

As mentioned, the NSA made several different modifications to the Lucifer cipher before it was accepted as the DES. Some modifications, like changing the S-Boxes to protect against differential cryptanalysis, were designed to improve the security of the cipher. Others, like changing the key length from 128-bits to 56-bits, may have been designed to ensure that the NSA still had the ability to break the cipher if needed.

Ultimately, Lucifer, or the DES, was not broken due to cryptographic vulnerabilities, but instead by a brute force attack that took advantage of the limited key space. DES was first broken in 1997, acting as the catalyst for the advent of the Advanced Encryption Standard.

Types of Cryptography

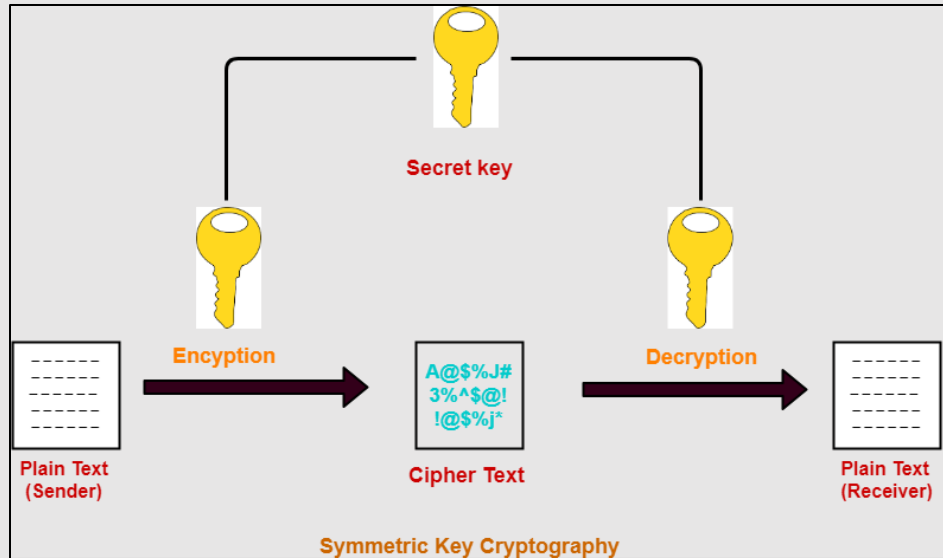
In general there are two types of cryptography:

Symmetric Key Cryptography:



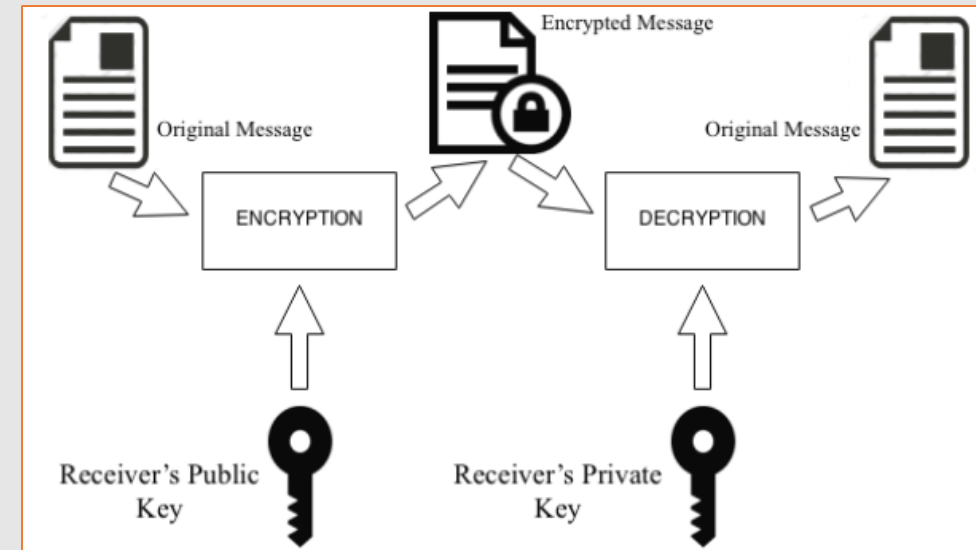
It is an encryption system where the sender and receiver of message use a single common key to encrypt and decrypt messages. Symmetric Key Systems are faster and simpler but the problem is that sender and receiver have to somehow exchange key in a secure manner. The most popular symmetric key cryptography system is Data Encryption System(DES).

Asymmetric Key Cryptography: Under this system a pair of keys is used to encrypt and decrypt information. A public key is used for encryption and a private key is used for decryption. Public key and Private Key are different. Even if the public key is known by everyone the intended receiver can only decode it because he alone knows the private key .



Symmetric Key Cryptography

Asymmetric Key Cryptography

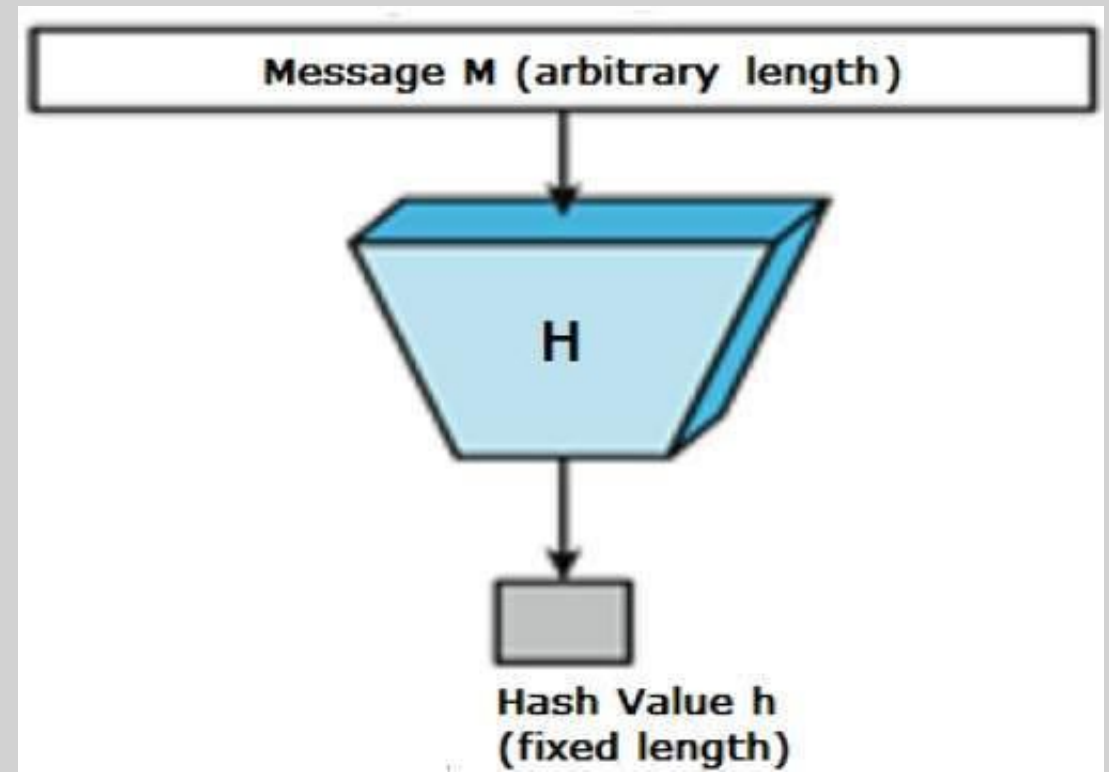


Hash Function

Hash functions are extremely useful and appear in almost all information security applications.

A hash function is a mathematical function that converts a numerical input value into another compressed numerical value. The input to the hash function is of arbitrary length but output is always of fixed length.

Values returned by a hash function are called message digest or simply hash values.



Features of Hash Functions :

The typical features of hash functions are –

Fixed Length Output (Hash Value)

Hash function converts data of arbitrary length to a fixed length. This process is often referred to as hashing the data.

In general, the hash is much smaller than the input data, hence hash functions are sometimes called compression functions.

Since a hash is a smaller representation of a larger data, it is also referred to as a digest.

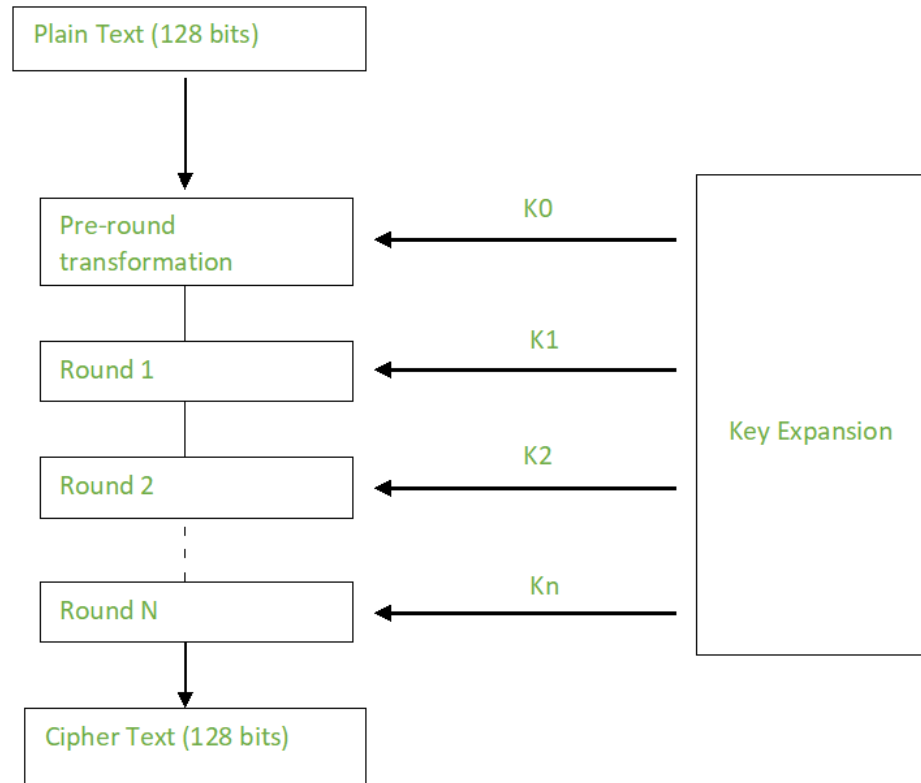
Hash function with n bit output is referred to as an n -bit hash function. Popular hash functions generate values between 160 and 512 bits.

Efficiency of Operation

Generally for any hash function h with input x , computation of $h(x)$ is a fast operation.

Computationally hash functions are much faster than a symmetric encryption.

AES

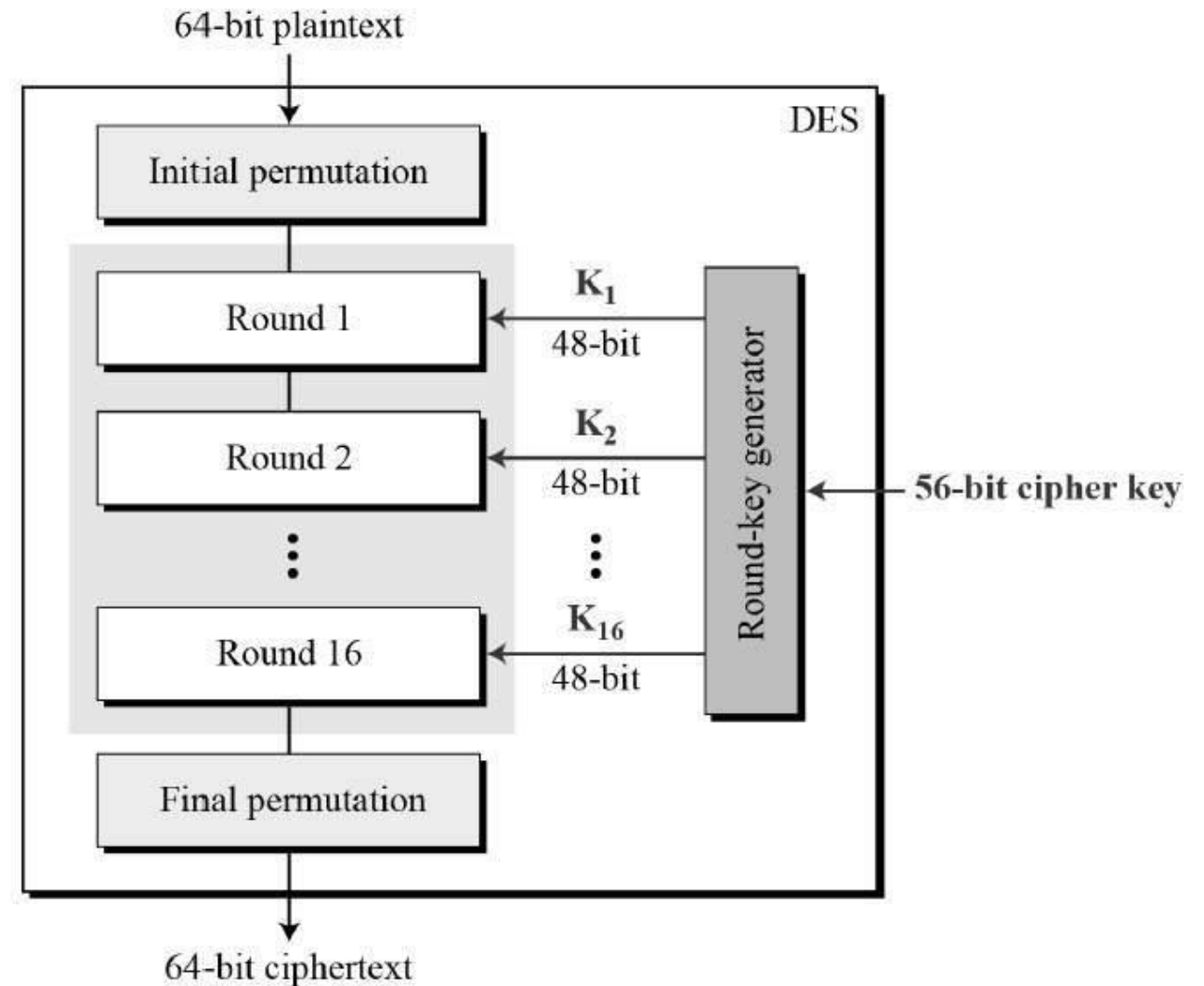


The AES Encryption algorithm (also known as the Rijndael algorithm) is a symmetric block cipher algorithm with a block/chunk size of 128 bits. It converts these individual blocks using keys of 128, 192, and 256 bits. Once it encrypts these blocks, it joins them together to form the ciphertext.

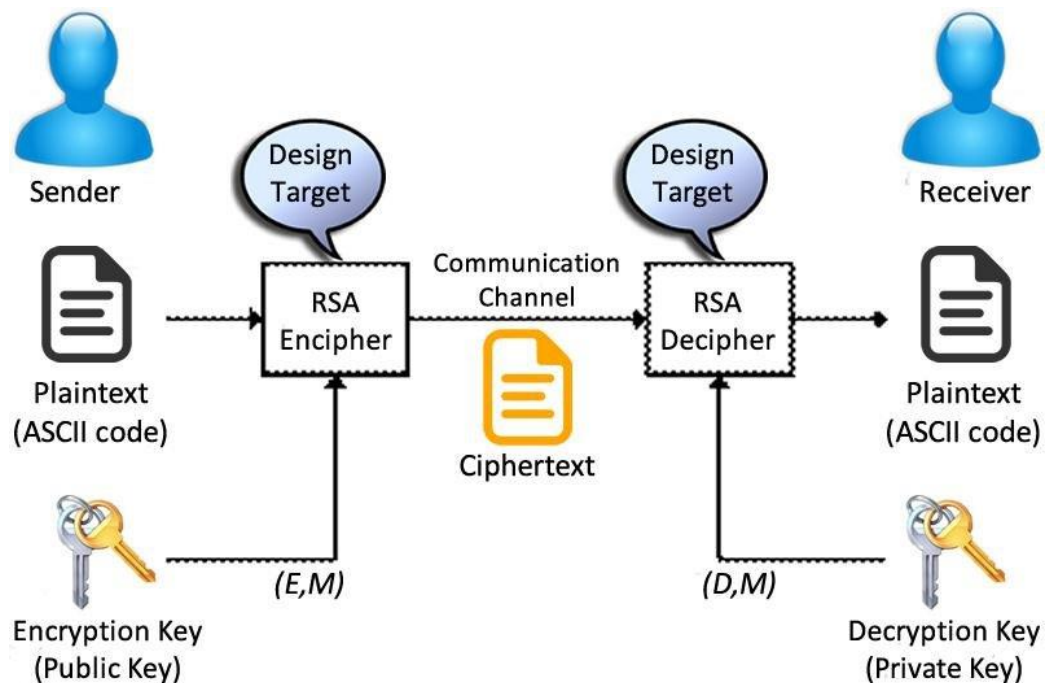
It is based on a substitution-permutation network, also known as an SP network. It consists of a series of linked operations, including replacing inputs with specific outputs (substitutions) and others involving bit shuffling (permutations).

Data Encryption Standard

The Data Encryption Standard (DES) is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST). DES is an implementation of a Feistel Cipher. It uses 16 round Feistel structure. The block size is 64-bit. Though, key length is 64-bit, DES has an effective key length of 56 bits, since 8 of the 64 bits of the key are not used by the encryption algorithm (function as check bits only). General Structure of DES is depicted in the following illustration –



RSA



RSA (Rivest–Shamir–Adleman) is an algorithm used by modern computers to encrypt and decrypt messages. It is an asymmetric cryptographic algorithm. Asymmetric means that there are two different keys. This is also called public key cryptography, because one of the keys can be given to anyone. The other key must be kept private. The algorithm is based on the fact that finding the factors of a large composite number is difficult: when the factors are prime numbers, the problem is called prime factorization. It is also a key pair (public and private key) generator.

RSA involves a public key and private key. The public key can be known to everyone- it is used to encrypt messages. Messages encrypted using the public key can only be decrypted with the private key. The private key needs to be kept secret. Calculating the private key from the public key is very difficult.

Shamir's Secret Sharing (SSS)

Shamir's Secret Sharing (SSS) is a key distribution algorithm. It is named for the well-known Israeli cryptographer Adi Shamir who co-invented the Rivest–Shamir–Adleman (RSA) algorithm. SSS divides a secret, such as a cryptokey, into parts called shares. The shares are distributed to a group of people who are parties to the conversation. The parts of the secret are brought together to reconstruct the secret, but an important feature of Shamir's Secret Sharing is that the total number of shares is not needed to reconstruct the secret. A number less than the total number, called the threshold, is required. This helps avoid failures in decrypting the closely-held information should just one or a few parties be unavailable. SSS is practical in its solution to the key-sharing problems many arrangements face, and is therefore usually used to secure the keys to something that is encrypted or secure using other tools or algorithms. A simple illustration of SSS is that of a vault that only a corporate board may access. The passcode is encrypted by SSS, so a quorum (threshold) of board members is needed to authorize the display or release of the vault passcode. If a board member is traveling, but the threshold is met, SSS still allows for a reasonable assurance that the vault is secure.

$$\begin{aligned} f(x) &= \sum_{j=0}^2 y_j \cdot \ell_j(x) \\ &= y_0 \ell_0(x) + y_1 \ell_1(x) + y_2 \ell_2(x) \\ &= 1942 \left(\frac{1}{6}x^2 - \frac{3}{2}x + \frac{10}{3} \right) + 3402 \left(-\frac{1}{2}x^2 + \frac{7}{2}x - 5 \right) + 4414 \left(\frac{1}{3}x^2 - 2x + \frac{8}{3} \right) \\ &= 1234 + 166x + 94x^2 \end{aligned}$$

Our Proposed Scheme

The humble motive of our study is to express SSS using Public key. Threshold cryptography enables the construction of reliable and robust key management system which can reconstruct the key even in the destruction of some shares and on the contrary the key cannot be reconstructed unless a predefined set of shares has been accumulated.

Algorithm for Share Generation:

Encryption Phase:

Step – 1: Sender finds the digest of the encrypted key using MD5 Hash algorithm and generate 128 bit (16 byte) secret key.

Step – 2: Generate required number of shares or mask using the (n, k) Share generation algorithm.

Step – 3: Next convert secret plaintext into a corresponding binary value. (User will input the plain text or read it from an existing file.).

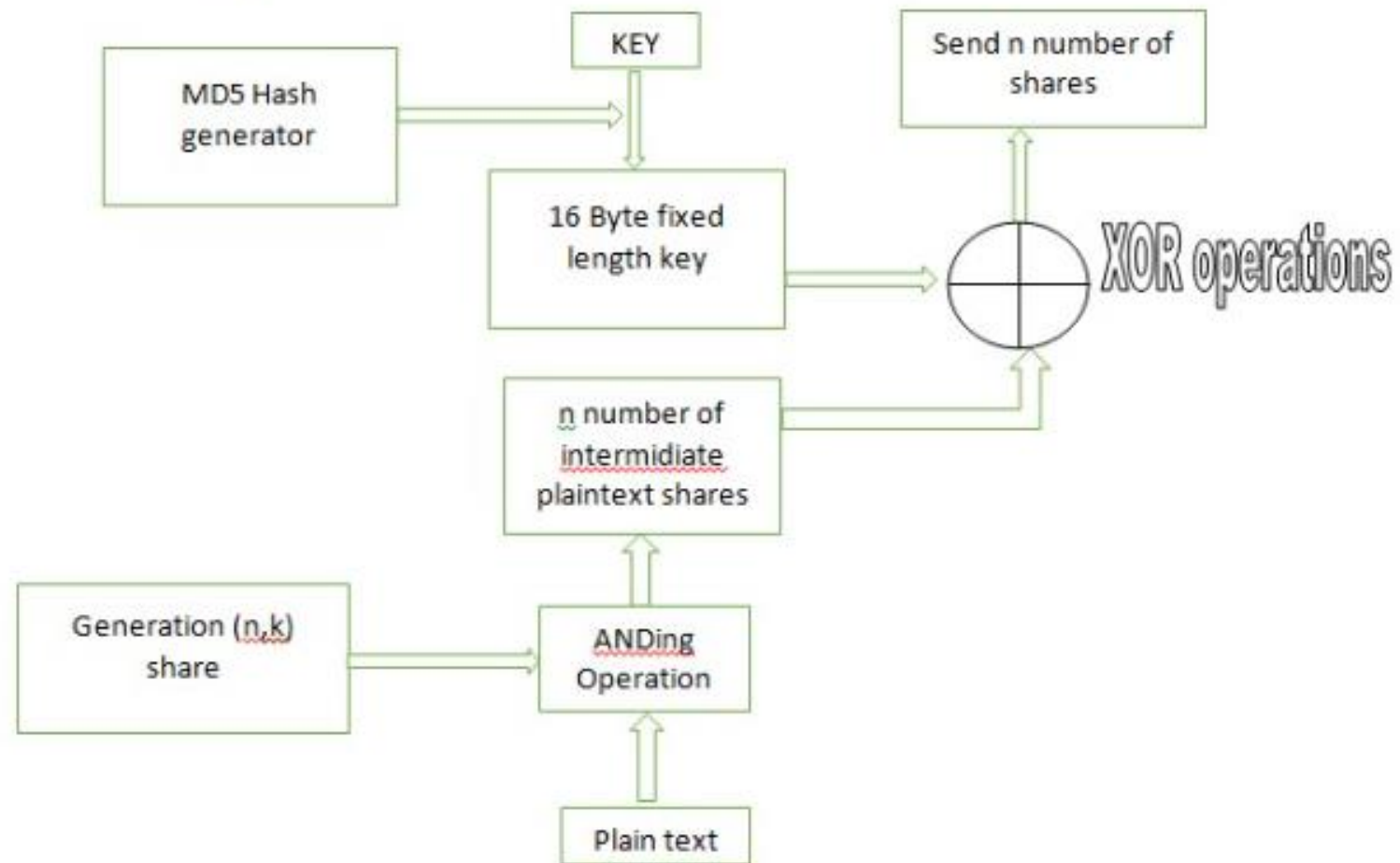
Step – 4: Select the mask for the first share and go on ANDing with the PLAIN TEXT up to the last bit, which generates the first intermediate share.

Step – 5: Repeating the step-4 with every share generated in masking process which generates the intermediate shares.

Step – 6: Select the first intermediate share and XOR it with the 16 byte secret key obtained from step 2 for generating the first final share.

Step – 7: Repeat the process of step-6 with every intermediate share in order to get the final share.

Flow chart Encryption Phase:



Decryption Phase:

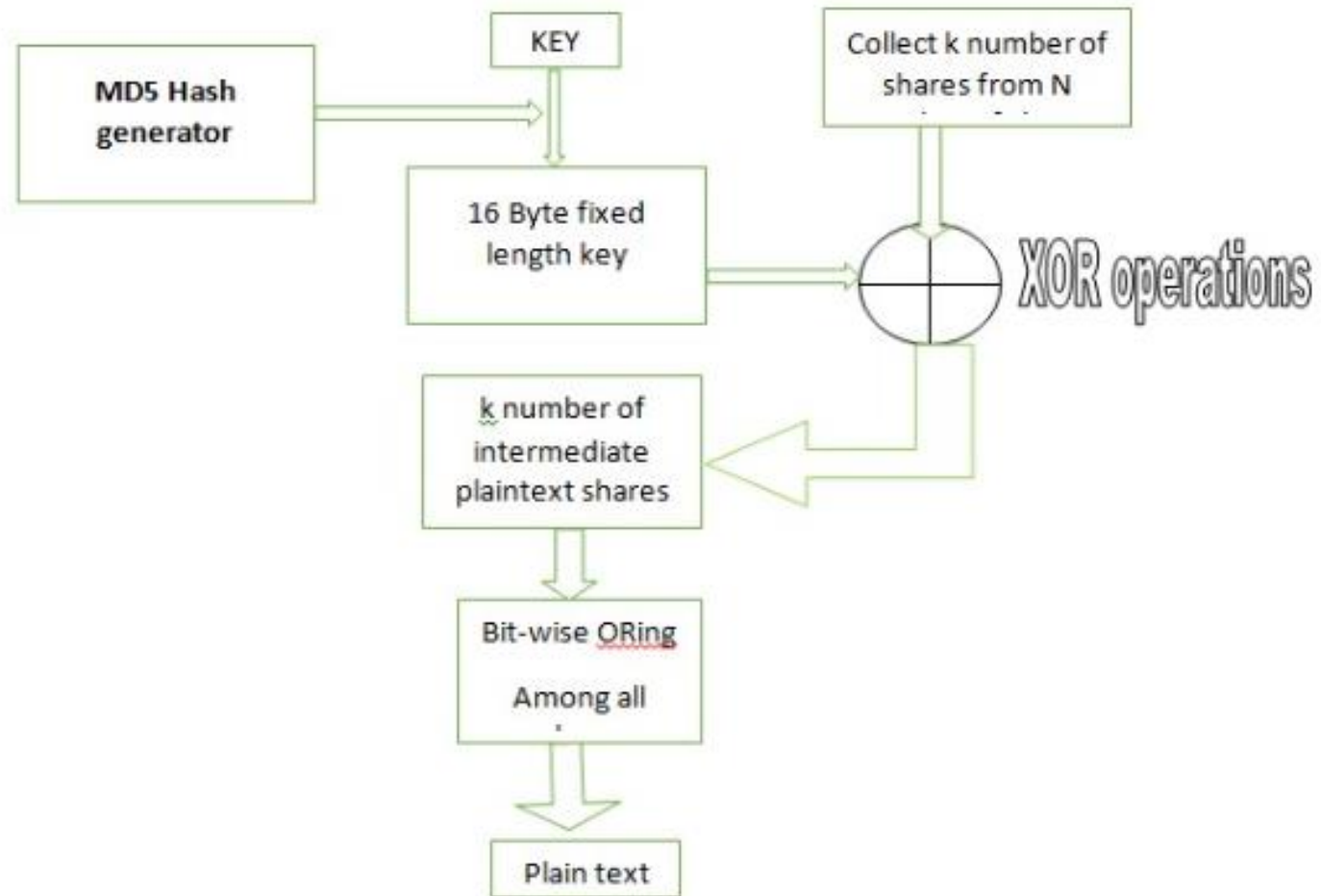
Step – 1: Sender finds the digest of the encrypted key using MD5 Hash algorithm and generate 128 bit (16 byte) secret key.

Step – 2: Select the minimum required k number of shares or mask from n shares. (Note: If Sender did not select minimum k number of share from n shares then it will lead towards data lost.)

Step – 3: Select k number of shares from step 2 and XOR it with the 16 byte secret key from step 1 for generate the intermediate shares.

Step – 4: Select all intermediate share generated in step 3 and bit-wise ORing for retrieving the plain text.

Flow chart Decryption Phase:



Algorithm for Share Generation:

Here we are presenting the algorithm for designing the masks for n shares with threshold k .

Step-1:

List all row vectors of size n having the combination of $(k-1)$ numbers of 0's and $(n-k+1)$ numbers of 1's and arrange them in the form of a matrix. Obvious dimension of the matrix will be $nC_{k-1} \times n$.

Step-2:

Transpose the matrix generated in Step-1. Obvious dimension of the transposed matrix will be $n \times nC_{k-1}$. Each row of this matrix will be the individual mask for n different shares. The size of each mask is nC_{k-1} bits, i.e. the size of the mask varies with the value of n and k . (It may be noted that the masking patterns are not unique. Different arrangements of the row vectors in Step-1 leads to different sets of masks but for a particular set, the masks are unique and they satisfy the requirements.)

Let us consider the previous example where $n=5$ and $k=3$.

- **Step-I:**

List of row vectors of size 5 bits with 2 numbers of 0's and 3 numbers of 1's.

1	1	1	0	0
1	1	0	1	0
1	1	0	0	1
1	0	1	1	0
1	0	1	0	1
1	0	0	1	1
0	1	1	1	0
0	1	1	0	1
0	1	0	1	1
0	0	1	1	1

Dimension of the matrix is $5C2 \times 5$ i.e. 10×5 Step-2: Take the transpose of the above matrix and we get the desired masks for five shares as listed above in the form of matrix of dimension $5 \times 5C2$ i.e. 5×10 . There are five masks each of size 10 bits.

- Step-II:

Transpose the matrix of step-1 shown below-

1	1	1	1	1	1	0	0	0	0
1	1	1	0	0	0	1	1	1	0
1	0	0	1	1	0	1	1	0	1
0	1	0	1	0	1	1	0	1	1
0	0	1	0	1	1	0	1	1	1



Conclusion

As we toward a society where automated information resources are increased and cryptography will continue to increase in importance as a security mechanism. Electronic networks for banking, shopping, inventory control, benefit and service delivery, information storage and retrieval, distributed processing, and government applications will need improved methods for access control and data security. The information security can be easily achieved by using Cryptography technique. DES is now considered to be insecure for some applications like banking system. there are also some analytical results which demonstrate theoretical weaknesses in the cipher. So it becomes very important to augment this algorithm by adding new levels of security to make it applicable. By adding additional key, modified S-Box design, modifies function implementation and replacing the old XOR by a new operation as proposed by this thesis to give more robustness to DES algorithm and make it stronger against any kind of intruding. DES Encryption with two keys instead of one key already will increase the efficiency of cryptography.

REFERENCES

[1] N. Sharma , Prabhjot and H. Kaur, "A Review of Information Security using Cryptography Technique," International Journal of Advanced Research in Computer Science, vol. 8, no. Special Issue, pp. 323-326, 2017.

[2] B. Preneel, Understanding Cryptography: A Textbook for Students and Practitioners, London: Springer, 2010.

[3] J. Katz and Y. Lindell, Introduction to Modern Cryptography, London: Taylor & Francis Group, LLC , 2008.

[4] S. J. Lincke and A. Hollan, "Network Security: Focus on Security, Skills, and Stability," in 37th ASEE/IEEE Frontiers in Education Conference, Milwaukee, 2007.

[5] O. O. Khalifa, M. R. Islam, S. Khan and M. S. Shebani, "Communications cryptography," in RF and Microwave Conference, 2004. RFM 2004. Proceedings, Selangor, 2004.

[6] N. Jirwan, A. Singh and S. Vijay , "Review and Analysis of Cryptography Techniques," International Journal of Scientific & Engineering Research, vol. 3, no. 4, pp. 1-6, 2013 .

[7] S. Tayal, N. Gupta, P. Gupta, D. Goyal and M. Goyal, "A Review paper on Network Security and Cryptography," Advances in Computational Sciences and Technology , vol. 10, no. 5, pp. 763-770, 2017.

[8] A. Gupta and N. K. Walia, "Cryptography Algorithms: A Review," INTERNATIONAL JOURNAL OF ENGINEERING DEVELOPMENT AND RESEARCH, vol. 2, no. 2, pp. 1667-1672, 2014.

REFERENCES

[9] J. Callas, "The Future of Cryptography," Information Systems Security, vol. 16, no. 1, pp. 15-22, 2007.

[10] J. L. Massey, "Cryptography—A selective survey," Digital Communications, vol. 85, pp. 3-25, 1986.

[11] B. Schneier, "The Non-Security of Secrecy," Communications of the ACM, vol. 47, no. 10, pp. 120-120, 2004.

[12] N. Varol, F. Aydoğan and A. Varol, "Cyber Attacks Targetting Android Cellphones," in The 5th International Symposium on Digital Forensics and Security (ISDFS 2017), Tirgu Mures, 2017.

[13] K. Chachapara and S. Bhadlawala, "Secure sharing with cryptography in cloud," in 2013 Nirma University International Conference on Engineering (NUICONE), Ahmedabad, 2013.

[14] H. Orman, "Recent Parables in Cryptography," IEEE Internet Computing, vol. 18, no. 1, pp. 82-86, 2014.

[15] R. GENNARO, "IEEE Security & Privacy," IEEE Security & Privacy, vol. 4, no. 2, pp. 64 - 67, 2006.

[16] B. Preneel, "Cryptography and Information Security in the Post-Snowden Era," in IEEE/ACM 1st International Workshop on TEchnical and LEgal aspects of data pRivacy and SEcurity, Florence, 2015.

[17] S. B. Sadkhan, "Cryptography : current status and future trends," in International Conference on Information and Communication Technologies: From Theory to Applications, Damascus, 2004.

[18] F. Piper and S. Murphy, Cryptography: A Very Short Introduction, London: Oxford University Press, 2002.

[19] J. P. Aumasson, SERIOUS CRYPTOGRAPHY A Practical Introduction to Modern Encryption, San Francisco: No Starch Press, Inc, 2018 .

REFERENCES

[20] J. F. Dooley, A Brief History of Cryptology and Cryptographic Algorithms, New York: Springer, 2013.

[21] T. S. Denis and S. Johnson, Cryptography for Developers, Boston: Syngress Publishing Inc, 2007 .

[22] W. D. A. M. E. HELLMAN, "New directions in cryptography," IEEE Transactions on Information Theory, Vols. IT-22, no. 6, pp. 644-654, 1976.

[23] W. Stallings, Cryptography and Network Security Principles and Practices, New York: Prentice Hall, 2005.

[24] A. Shamir: "How to share a secret?" Comm ACM, 22(11):612-613, 1979.

[25] G. Blakley : "Safeguarding cryptographic keys " Proc. of AFIPS National Computer Conference, 1979.

[26] C. Asmuth and J. Bloom : "A modular approach to key safeguarding" IEEE transaction on Information Theory, 29(2):208-210, 1983.

[27] Y. Desmedt "Some recent research aspects of threshold cryptography" Proc. of ISW'97 1st International Information Security Workshop vol.1196 of LNCS paper 158-173 Springer-Verlag 1997.

[28] Y. Desmedt and Y. Frankel "Threshold cryptosystems" Proc. of CRYPTO'89 volume 435 of LNCS, paper 307-315 Springer Verlag 1990.

[29] Y. Desmedt and Y. Frankel "Shared generation of authenticators and signatures" Proc. of CRYPTO'91 volume 576 of LNCS pages 457-469 Springer Verlag 1992.

[30] Y. Desmedt and Y. Frankel "Homomorphic zero knowledge threshold schemes over any finite abelian group" SIAM journal on Discrete Mathematics 7(4): 667- 675, 1994.

[31] H. F. Hua ng and C.C. Chang "A novel efficient (t, n) threshold proxy signature scheme" Information Sciences 176(10): 1338-1349, 2006.

[32] A. De Santis, Y. Desmedt, Y. Frankel and M. Yung "How to share a function securely?" In proc. of STOC 94, paper 522-533, 1994.

REFERENCES

[33] V. Shoup “Practical threshold signatures” In Proc of Eurocrypt 2000. volume 1807 of LNCS paper 207-220, Springer-Verlag 2000.

[34] Bozkurt, Kaya, Selcuk, Guloglu “Threshold Cryptography Based on Blakely Secret Sharing” Information Sciences.

[35] K. Kaya and A. A. Selcuk “Threshold Cryptography based on Asmuth-Bloom Secret Sharing” Information Sciences 177(19) 4148-4160, 2007