USN | | | | | | | | | | |

CMRIT — CMR INSTITUTE OF TECHNOLOGY, BENGALURU. ACCREDITED WITH A++ GRADE BY NAAC

INTERNAL ASSESSMENT TEST – III

Sub:	CRYPTOGRAPHY							Code:	18EC744
Date:	27/ 12 / 2022	Duration:	90 mins	Max Marks:	50	Sem:	VII	Branch:	ECE

Answer any 5 full questions

		Marks	CO	RBT
1	Perform encryption using RSA algorithm for p=5, q = 11, e = 3, m = 9.	[10]	CO2	L3
2	What is Diffie Hellman key exchange algorithm? Describe how the secret is computed by Alice and Bob to encrypt and decrypt the information.	[10]	CO2	L3
3	With a neat diagram, explain public-key cryptosystem secrecy and Authentication.	[10]	CO2	L2
4.	List out different types of LFSR-based Keystream generator.	[10]	CO5	L1
5.	Explain the following with necessary diagrams: a. Generalized Geffe Generator b. Threshold Generator	[10]	CO5	L2
6.	Explain Additive Generators. Also explain fish and pike Additive Generator.	[10]	CO5	L2
7.	With a neat diagram, explain the concept of Gifford.	[10]	CO5	L2

IAT-III Scheme of solutions

Q. no.	Questions	Marks
1.	Perform encryption using RSA algorithm for p=5, q = 11, e = 3, m = 9.	10M

	$n = pq = 5 \times 11 = 55$ $\emptyset(n) = (p-1) \times (q-1) = 4 \times 10 = 40$ $e = 3 \;\; and \;\; m = 9$ $ed \; mod \; \emptyset(n) \equiv 1 => d = e^{-1} \; mod \; \emptyset(n) => d = 3^{-1} \; mod \; 40 => d = -13 \; mod \; 40 = \mathbf{27}$	

q	r_1	r_2	r	t_1	t_2	$t = t_1 - qt_2$
13	40	3	1	0	1	−13
3	3	1	0	1	− 13	40
	1	0		**−13**	40	

$PU = \{3,55\} \;\; and \;\; PR = \{27,55\}$

$C = M^e \; mod \; n => C = 9^3 mod \; 55 = 14$

$M = C^d \; mod \; n = 14^{27} \; mod \; 55 = 9$

$14^{27} \; mod \; 55$

$(27)_{10} = (11011)_2$

$1: 14 \; mod \; 55 = 14$

$1: (14)^2 \times 14 \; mod \; 55 = 49$

$0: (49)^2 mod \; 55 = 36$

$1: (36)^2 \times 14 \; mod \; 55 = 49$

$1: (49)^2 \times 14 \; mod \; 55 = 9$

2. What is Diffie Hellman key exchange algorithm? Describe how the secret is computed by Alice and Bob to encrypt and decrypt the information. **10M**

Diffie Hellman Key Exchange Algorithm:

1. In this scheme, there are two publicly known numbers those are: a prime number q and an integer α that is a primitive root of q.
2. User A selects a random integer $X_A < q$ and compute $Y_A = \alpha^{X_A} \; mod \; q$.
3. User B selects a random integer $X_B < q$ and compute $Y_B = \alpha^{X_B} \; mod \; q$.
4. User A computes the key as $K_A = {Y_B}^{X_A} \; mod \; q$
5. User B computes the key as $K_B = {Y_A}^{X_B} \; mod \; q$

$K_A = {Y_B}^{X_A} \; mod \; q$

$K_A = (\alpha^{X_B} \; mod \; q)^{X_A} \; mod \; q$

$K_A = (\alpha^{X_B})^{X_A} \; mod \; q$

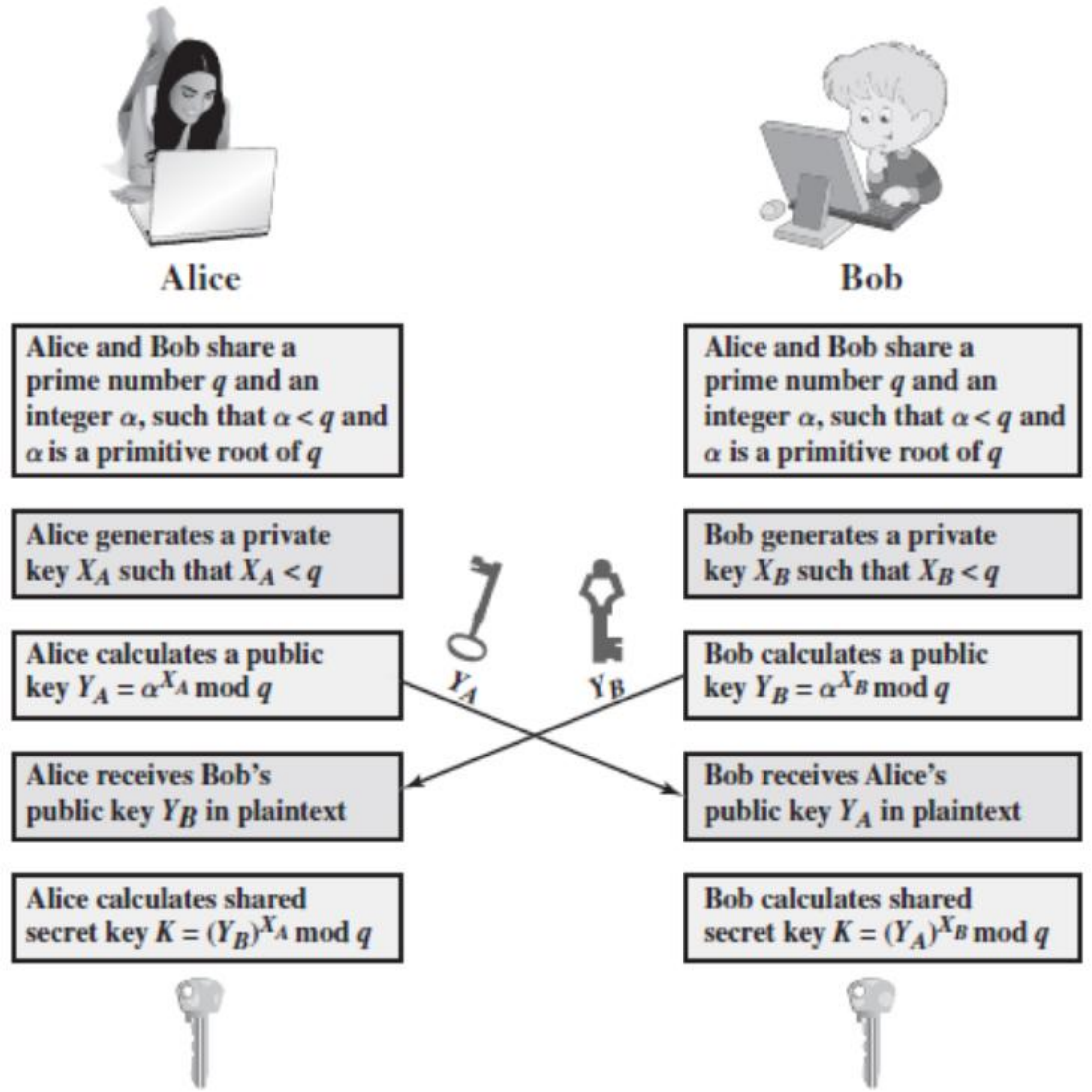
$K_A = \alpha^{X_B X_A} \; mod \; q$

$K_A = (\alpha^{X_A})^{X_B} \; mod \; q$

$K_A = (\alpha^{X_A} \; mod \; q)^{X_B} \; mod \; q$

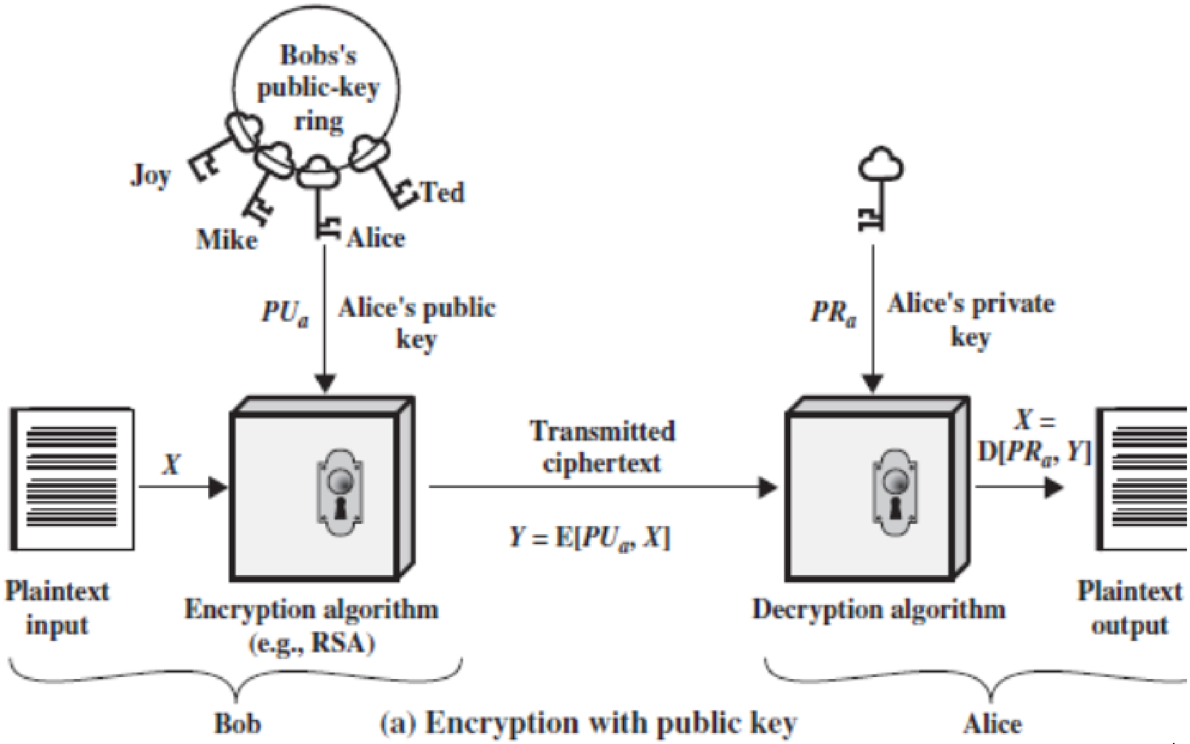
$K_A = {Y_A}^{X_B} \; mod \; q$

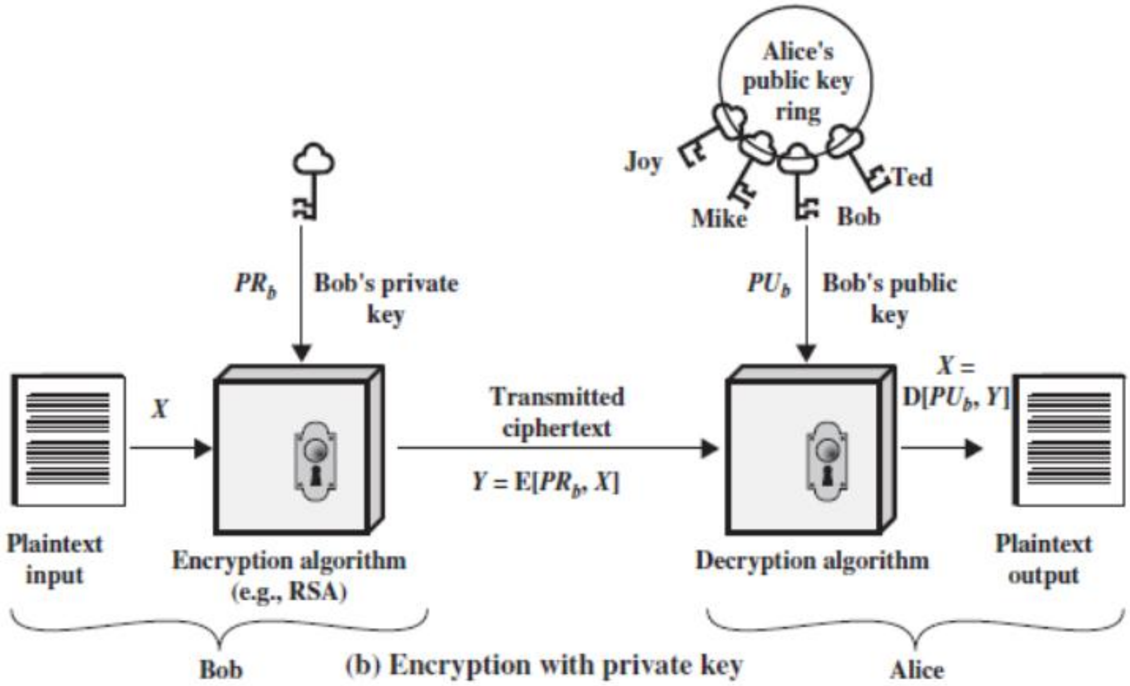
$K_A = K_B$

	 The Diffie-Hellman Key Exchange	
3.	With a neat diagram, explain public-key cryptosystem secrecy and Authentication. Essential steps in public key cryptosystem: 1. Each user generates a pair of key to be used for encryption and decryption. 2. Each user place one of the key in public register and other one is kept private. Each user maintains a collection of public keys obtained from others. 3. If Bob wants to send a confidential message to Alice, Bob encrypts the message using Alice's public key. 4. When Alice receives the message, it decrypts the message using its private key. 5. As long as the user's private key is protected the communication is secure. At any time a system can change its private key and publish the companion public key to replace its old public key 6. The key used in symmetric key is named as secret key and the 2 keys used in public key cryptography are named as public key and private key.	10M

$Notation\ Used:\ K_a = Secret\ key\ of\ sender\ 'A'$

$PU_a = Public\ key\ of\ sender\ 'A'$ $PR_a = Private\ key\ of\ se$



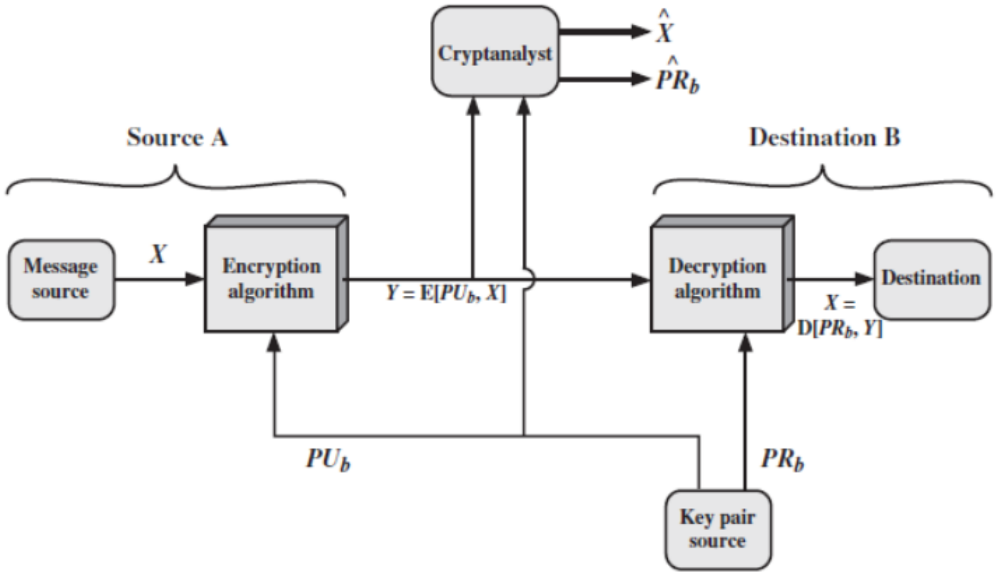


Public Key Cryptosystem-Secrecy:

1. Source 'A' sends the plaintext $X = [X1, X2, \ldots \ldots Xm]$. The m element of X is some alphabet in
the message.
2. As the message is intended for user 'B' , 'B' generates 2 keys
a) Private Key (PRb)
b) Public Key (PUb) and PUb is publicly available so that it is accessible by A.
3. With the message X and encryption key PUb, sender forms the cipher text
$Y = [Y1, Y2, \ldots \ldots YN]$ $where\ Y = E(PUb, X)$
4. At the receiver, the intended receiver matches the key and find the message

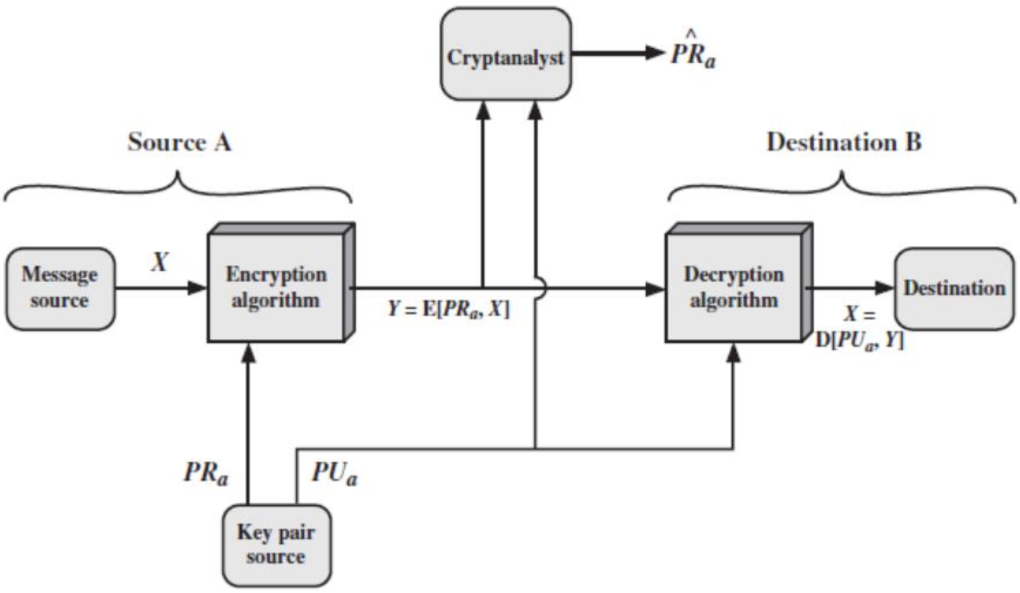
$X = D(PRb, Y)$

5. It is assumed that the cryptanalysts have the knowledge of encryption $(E)$and decryption (D)algorithms. If the cryptanalyst is interested only in this particular message, then its focus is to recover X, by generating a plaintext estimate $\hat{X}$. But if Cryptanalyst is interested in being able to read future message as well, it will try to recover PRb, by generating an estimate $\hat{PR}b$.
6. Either of the 2 keys can be used for encryption, with other being used for decryption. This above scheme provides confidentiality.
7. As anybody can encrypt the message using $B's$ public key and claim to be came from $'A'$



Public-Key Cryptosystem: Secrecy

Public Key Cryptosystem-Authentication:



Public-Key Cryptosystem: Authentication

1. If $'A'$ wants to communicate to $'B'$, then $'A'$ encrypt the message using $A's$ private key.
2. $'B'$ can decrypt the message using $A's$ public key.
3. As message was encrypted using $A's$ private key, only $'A'$ could prepare the message. This entire message serves as a digital signature.
4. It is important to alter the message without access to $A's$ private key. So this message is authenticated both in terms of source and data integrity.
5. The encryption and decryption can be represented as :

$Y = E(PRa, X)$

$X = D(PUa, Y)$

6. This public key encryption doesn't provide confidentiality because all will have $A's$ public key hence can decrypt the message easily.

7. It is safe from alteration but not from eavesdropping.

4. List out different types of LFSR-based Keystream generator. **10M**

The list of LFSR based keystream generators are:

a) Geffe Generator
b) Generalized Geffe Generator
c) Jennings Generator
d) Beth-Piper Stop-and-Go Generator
e) Alternating Stop-and-Go Generator
f) Bilateral Stop-and-go Generator
g) Threshold Generator
h) Self-Decimated Generator
i) Multispeed Inner-Product Generator
j) Summation Generator
k) DNRSG (dynamic random-sequence generator)
l) Gollmann Cascade
m) Shrinking Generator
n) Self-Shrinking Generator

With Description of each.

5. Explain the following with necessary diagrams: **10M**

a. Generalized Geffe Generator
b. Threshold Generator

a. **Geffe Generator:**

This generator uses three LFSRs, combined in a nonlinear manner. Two of the LFSRs are input a multiplexer and the third LFSR controls the output of the multiplexer. If $a1$, $a2$ *and* $a3$ are the output of the three LFSRs, the output of the Geffe generator can be represented as: $b = (a1 \wedge a2) \oplus ((\neg a1 \wedge a3))$**.** If the LFSR have length $n1$, $n2$ *and* $n3$ respectively, then the linear complexity is: $(n1 + 1)n2 + n1n3$

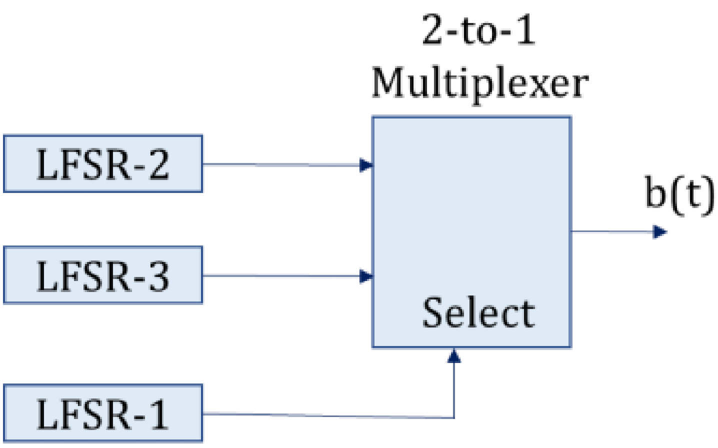


Figure: Geffe generator

Although this generator looks good on paper, it is cryptographically weak and falls to a correlation attack.

b) Generalized Geffe Generator:

Instead of choosing between two LFSRs, this scheme chooses between k LFSRs, where k is power of 2. There are $k + 1$ LFSRs total. LFSR-1 must be clocked log2 k times faster than the other k LFSRs. Though this scheme is complex than Gaffe generator, same kind of correlation attack is possible.

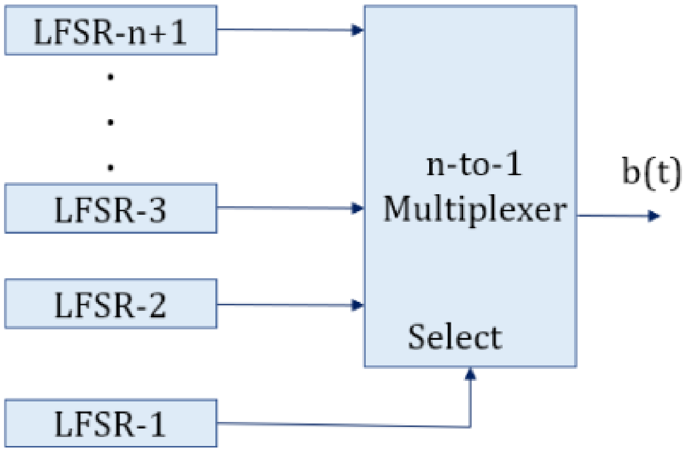


Figure: Generalized Geffe generator

6. Explain Additive Generators. Also explain fish and pike Additive Generator. **10M**

ADDITIVE GENERATORS:

• *Additive generators* are extremely efficient because they produce random words instead of random bits. They are not secure on their own, but can be used as building blocks or secure generators.

• The initial state of the generator is an array of n-bit words: 8-bit words, 16-bit words, 32-bit words. The initial state is the key. The *i*th word of the generator is

$X_i = (X_{i-a} + X_{i-b} + X_{i-c} + \cdots + X_{i-m}) \bmod 2n$

• If the coefficients a, b, c, … m are chosen right, the period of this generator is at least $2n - 1$

Example: (55,24,0) is a primitive polynomial mod 2. This means that the following additive generator is maximal length.

$X_i = (X_{i-55} + X_{i-24}) \bmod 2n$

This works because, primitive polynomial has three coefficients. If it has more coefficient, then we need some additional requirements to make it maximal length.

Fish:

➢ Fish is an additive generator based on techniques used in the shrinking generator. It produces a stream of 32-bit words which can be XORed with the plaintext stream to produce ciphertext, or XORed with ciphertext stream to produce plaintext.

➢ The algorithm is named as it is Fibonacci Shrinking generator.

➢ First, it uses two additive generators. The key is the initial values of these generators.

$A_i = (A_{i-55} + A_{i-24}) \bmod 232$

$B_i = (B_{i-52} + A_{i-19}) \bmod 232$

➢ These sequences are shrunk, as a pair, depending on the least significant bit of B_i: if it is 1, use the pair; if it is 0, ignore the pair.

➢ C_j is the sequence of used words from A_i and D_j is the sequence of used words from B_i. These words are used in pairs- C_{2j}, C_{2j+1}, D_{2j} and D_{2j+1}- to generate two 32-bit output words: K_{2j} and K_{2j+1}.

$E_{2j} = C_{2j} \oplus (D_{2j} \wedge D_{2j+1})$

$F_{2j} = D_{2j+1} \wedge (E_{2j} \wedge C_{2j+1})$

$K_{2j} = E_{2j} \oplus F_{2j}$

$K_{2i} = C_{2i+1} \oplus F_{2j}$

➢ This algorithm is fast, Unfortunately, it is also insecure; an attack has a work factor of about *240*.

Pike:

➢ Pike is the leaner, meaner version of Fish, developed by Ross Anderson, the man who broke Fish.

➢ It uses three additive generators. For example:

$A_i = (A_{i-55} + A_{i-24}) \bmod 232$

$B_i = (B_{i-57} + A_{i-7}) \bmod 232$

$C_i = (C_{i-58} + C_{i-19}) \bmod 232$

➢ To generate the keystream word, look at the additional carry bits.

➢ If all the three agree, then clock all three generators. If they don't, then just clock the two generators that agree. Save the carry bit for the next time. The final output is the XOR of the

	three generators. ➢ Pike is faster than Fish, as on average it requires 2.75 steps per output rather than 3 steps.	
7.	With a neat diagram, explain the concept of Gifford. • It was developed by David Gifford. It was used to encrypt news wire reports in Boston area from 1984 until 1988. • The algorithm has a single 8-byte register: $b0, b1, \dots \dots b7$. • The key is the initial stage of the register. • The algorithm works in OFB (output feedback); the plaintext doesn’t affect the algorithm at all. • To generate the key byte ki, concatenate $b0$ and $b2$ and concatenate $b4$ and $b7$. Multiply the two together to get a 32-bit number. The third byte from the left is ki. • To update the register, take $b1$and sticky right shift it 1 bit. (Sticky right shift: the left most bit is both shifted and also remains in place.). Take $b7$ and shift it 1 bit to the left; there should be a 0 in the right most bit position. Take the XOR of the modified $b1$, the modified $b7$ and $b0$. Shift the original register 1byte to the right and put this byte in the left most position. • This algorithm was broken in 1994. It concludes that, the feedback polynomial isn’t primitive and can be attacked.	10M

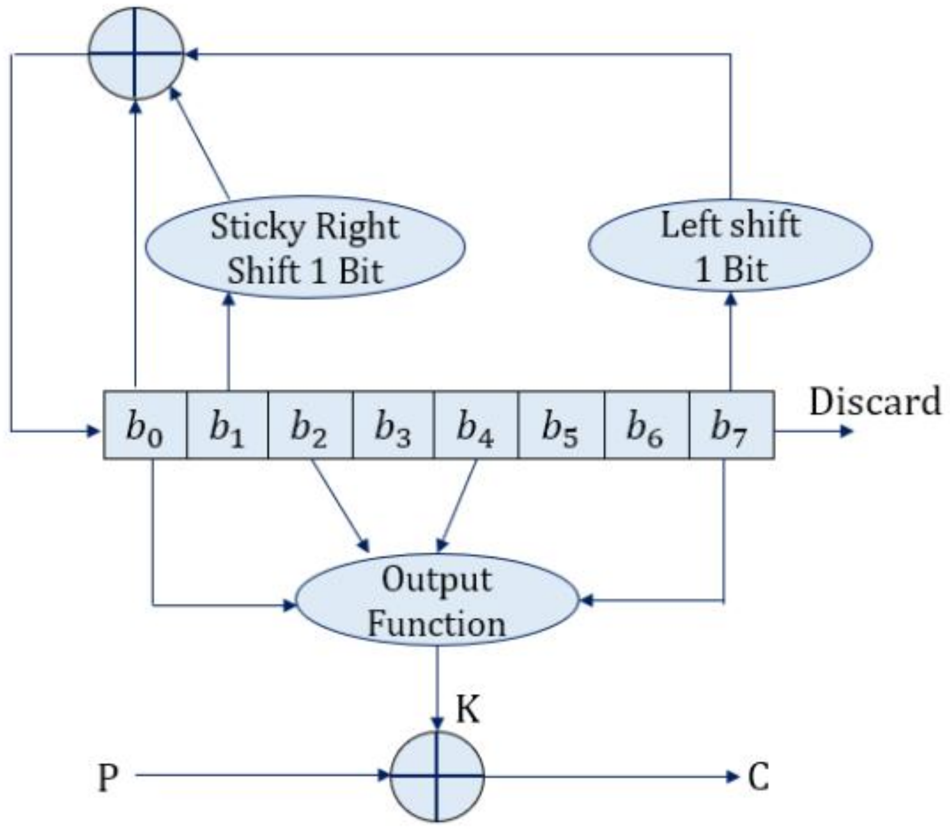


Figure: Gifford

CCI HOD