Sanidhya Kumar
20215138

CS304
Assignment 2

Page No.: 1/5/24 YOUVA
Date:

Q.1) (a) We need to prove that second row of S_4 can be obtained from the first row of S_4 using the following mapping –

$$(y_1, y_2, y_3, y_4) \rightarrow (y_2, y_1, y_4, y_3) \oplus (0, 1, 1, 0)$$

Lets check if 1st element of row 1 ($S_4[0][0]$) can be converted into 1st element of row 2 ($S_4[0][1]$).

→ $S_4[0][0] \Rightarrow \underbrace{0\ 1\ 1\ 1}_{[7]_{10}}$
$y_1\ y_2\ y_3\ y_4$

$\Rightarrow 1\ 0\ 1\ 1 \oplus 0\ 1\ 1\ 0$
$y_2\ y_1\ y_4\ y_3$

$\Rightarrow \underbrace{1\ 1\ 0\ 1}_{[13]_{10}} = S_4[1][0]$

→ Similarly, lets check if $S_4[0][1]$ can be converted into $S_4[1][1]$. using the mapping.

$S_4[0][1] = [13]_{10} \Rightarrow 1\ 1\ 0\ 1$

$\Rightarrow 1\ 1\ 1\ 0 \oplus 0\ 1\ 1\ 0$

$\Rightarrow 1\ 0\ 0\ 0 = [8]_{10} = S_4[1][1]$

→ Now, check next element

$S_4[0][2] = [14]_{10} \Rightarrow 1\ 1\ 1\ 0$

$\Rightarrow 1\ 1\ 0\ 1 \oplus 0\ 1\ 1\ 0$

$\Rightarrow 1\ 0\ 1\ 1 = [11]_{10} = S_4[1][2]$

→ We will check this transformation for all the elements of row 1.

2021511138

Savidhya Kumar

→ $S_4[0][3] = [3]_{10} \Rightarrow 0\ 0\ 1\ 1$

$\Rightarrow 0\ 0\ 1\ 1 \oplus 0\ 1\ 1\ 0$

$\Rightarrow 0\ 1\ 0\ 1 = [5]_{10} = S_4[1][3]$

→ $S_4[0][4] = [0]_{10} \Rightarrow 0\ 0\ 0\ 0$

$\Rightarrow 0\ 0\ 0\ 0 \oplus 0\ 1\ 1\ 0$

$\Rightarrow 0\ 1\ 1\ 0 = [6]_{10} = S_4[1][4]$

→ $S_4[0][5] = [6]_{10} \Rightarrow 0\ 1\ 1\ 0$

$\Rightarrow 1\ 0\ 0\ 1 \oplus 0\ 1\ 1\ 0$

$\Rightarrow 1\ 1\ 1\ 1 = [15]_{10} = S_4[1][5]$

→ $S_4[0][6] = [9]_{10} \Rightarrow 1\ 0\ 0\ 1$

$\Rightarrow 0\ 1\ 1\ 0 \oplus 0\ 1\ 1\ 0$

$\Rightarrow 0\ 0\ 0\ 0 = [0]_{10} = S_4[1][6]$

→ $S_4[0][7] = [10]_{10} \Rightarrow 1\ 0\ 1\ 0$

$\Rightarrow 0\ 1\ 0\ 1 \oplus 0\ 1\ 1\ 0$

$\Rightarrow 0\ 0\ 1\ 1 = [3]_{10} = S_4[1][7]$

→ $S_4[0][8] = [1]_{10} \Rightarrow 0\ 0\ 0\ 1$

$\Rightarrow 0\ 0\ 1\ 0 \oplus 0\ 1\ 1\ 0$

$\Rightarrow 0\ 1\ 0\ 0 = [4]_{10} = S_4[1][8]$

→ $S_4[0][9] = [2]_{10} \Rightarrow 0\ 0\ 1\ 0$

$\Rightarrow 0\ 0\ 0\ 1 \oplus 0\ 1\ 1\ 0$

$\Rightarrow 0\ 1\ 1\ 1 = [7]_{10} = S_4[1][9]$

→ $S_4[0][10] = [8]_{10} \Rightarrow 1\ 0\ 0\ 0$

$\Rightarrow 0\ 1\ 0\ 0 \oplus 0\ 1\ 1\ 0$

$\Rightarrow 0\ 0\ 1\ 0 = [2]_{10} = S_4[1][10]$

2021511138
Sanidhya Kumar

→ $S_4[0][11] = [5]_{10} \Rightarrow 0\ 1\ 0\ 1$

$\Rightarrow 1\ 0\ 1\ 0 \oplus 0\ 1\ 1\ 0$

$\Rightarrow 1\ 1\ 0\ 0 = [12]_{10} = S_4[1][11]$

→ $S_4[0][12] = [11]_{10} \Rightarrow 1\ 0\ 1\ 1$

$\Rightarrow 0\ 1\ 1\ 1 \oplus 0\ 1\ 1\ 0$

$\Rightarrow 0\ 0\ 0\ 1 = [1]_{10} = S_4[1][12]$

→ $S_4[0][13] = [12]_{10} \Rightarrow 1\ 1\ 0\ 0$

$\Rightarrow 1\ 1\ 0\ 0 \oplus 0\ 1\ 1\ 0$

$\Rightarrow 1\ 0\ 1\ 0 = [10]_{10} = S_4[1][13]$

→ $S_4[0][14] = [4]_{10} \Rightarrow 0\ 1\ 0\ 0$

$\Rightarrow 1\ 0\ 0\ 0 \oplus 0\ 1\ 1\ 0$

$\Rightarrow 1\ 1\ 1\ 0 = [14]_{10} = S_4[1][14]$

→ $S_4[0][15] = [15]_{10} \Rightarrow 1\ 1\ 1\ 1$

$\Rightarrow 1\ 1\ 1\ 1 \oplus 0\ 1\ 1\ 0$

$\Rightarrow 1\ 0\ 0\ 1 = [9]_{10} = S_4[1][15]$

∴ We proved that the entire second row of S_4 can be obtained from the first row using the mentioned mapping. Hence, proved!

(b) Any row of S_4 can be transformed into any other row by a similar type of operation. Let us try to find mapping which will transform row 2 to row 4.

Row 2 → Mapping → Row 4

20215138
Sowidhya Kumar

Page No.: YOUVA
Date:

Row 2

$(y_1, y_2, y_3, y_4) \rightarrow$ ~~$(z_1, z_2, z_3, z_4) \oplus (0, 1, 1, 0)$~~

$\rightarrow (x_1, x_2, x_3, x_4)$

Row 1

~~(z_1, z_2, z_3, z_4) are the mappings which would be used. Essentially they are some boolean fn on (y_1, y_2, y_3, y_4).~~ First, We will find correspondence from row 2 to 4. ie, from (y_1, y_2, y_3, y_4) to (x_1, x_2, x_3, x_4).

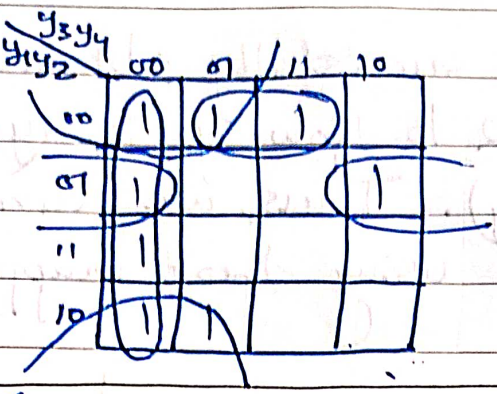
element	y_1	y_2	y_3	y_4	x_1	x_2	x_3	x_4	element
S[1][6]	0	0	0	0	1	1	0	1	S[3][6]
S[1][12]	0	0	0	1	1	1	0	0	S[3][12]
S[1][10]	0	0	1	0	0	1	0	1	S[3][10]
S[1][7]	0	0	1	1	1	0	0	0	S[3][7]
S[1][8]	0	1	0	0	1	0	0	1	S[3][8]
S[1][3]	0	1	0	1	0	1	1	0	S[3][3]
S[1][4]	0	1	1	0	1	0	1	0	S[3][4]
S[1][9]	0	1	1	1	0	1	0	0	S[3][9]
S[1][1]	1	0	0	0	1	1	1	1	S[3][1]
S[1][15]	1	0	0	1	1	1	1	0	S[3][15]
S[1][13]	1	0	1	0	0	1	1	1	S[3][13]
S[1][2]	1	0	1	1	0	0	0	0	S[3][2]
S[1][11]	1	1	0	0	1	0	1	1	S[3][11]
S[1][0]	1	1	0	1	0	0	1	1	S[3][0]
S[1][14]	1	1	1	0	0	0	1	0	S[3][14]
S[1][5]	1	1	1	1	0	0	0	1	S[3][5]

Now, we have the mapping. Lets construck K-Map in order to express x_i in terms of y_i for all i

Sanidhya Kumar
2021511138

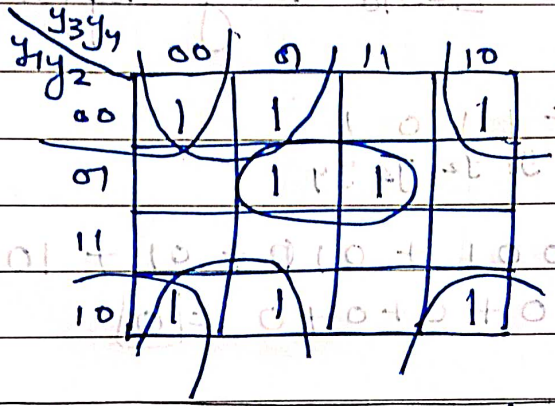
Here { + : Bitwise OR
· : Bitwise AND
$\bar{()}$: Complement

→ For x_1 :-



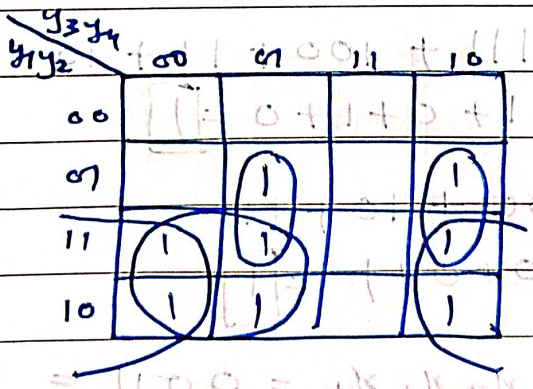
$$x_1 = (\bar{y_1}\,\bar{y_2}\,y_4) + (\bar{y_1}\,y_2\,\bar{y_4}) + (\bar{y_2}\,\bar{y_3}) + (\bar{y_3}\,\bar{y_4})$$

→ For x_2 :-



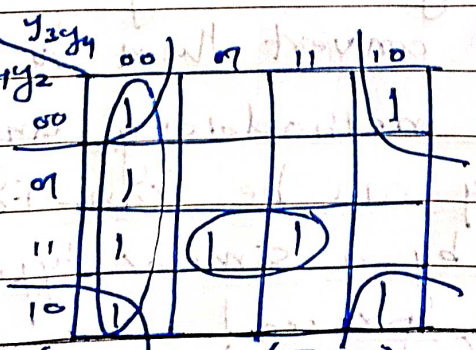
$$x_2 = (\bar{y_2}\,\bar{y_4}) + (\bar{y_1}\,y_2\,y_4) + (\bar{y_2}\,\bar{y_3})$$

→ For x_3 :-



$$x_3 = (y_2\,\bar{y_3}\,y_4) + (y_2\,y_3\,\bar{y_4}) + (y_1\,\bar{y_3}) + (y_1\,\bar{y_4})$$

→ For x_4 :-



$$x_4 = (\bar{y_2}\,\bar{y_4}) + (\bar{y_3}\,\bar{y_4}) + (y_1\,y_2\,y_4)$$

20215138
Samidhya Kumar

We, therefore, successfully obtained mappings from Row 2 to Row 4 ie, $(y_1\ y_2\ y_3\ y_4)$ to $(x_1\ x_2\ x_3\ x_4)$. Thus, we can convert row 2 to row 4 using these mappings.

Let us take an example and see if it works
Lets convert S[1][0] which is $[13]_{10}$ to S[3][0] which is $[3]_{10}$ using mappings we get

$[13]_{10} = 1101$
$y_1\ y_2\ y_3\ y_4$

$x_1 = 001 + 0101 + 01 + 10$
$= 0+0+0+0 = \boxed{0}$

$x_2 = 00 + 011 + 01$
$= 0+0+0 = \boxed{0}$

$x_3 = 111 + 100 + 11 + 10$
$= 1+0+1+0 = \boxed{1}$

$x_4 = 00 + 10 + 111$
$= 0+0+1 = \boxed{1}$

$\therefore x_1 x_2 x_3 x_4 = 0011 = [3]_{10}$

We can verify the same for all other elements of row 2 and convert them to row 4. Now, using this same methodology, ~~any~~ Similarly, any row of S_4 can be transformed into any other row by a similar type of transformation.

Hence, proved!

2021511138
Savidhya Kumar

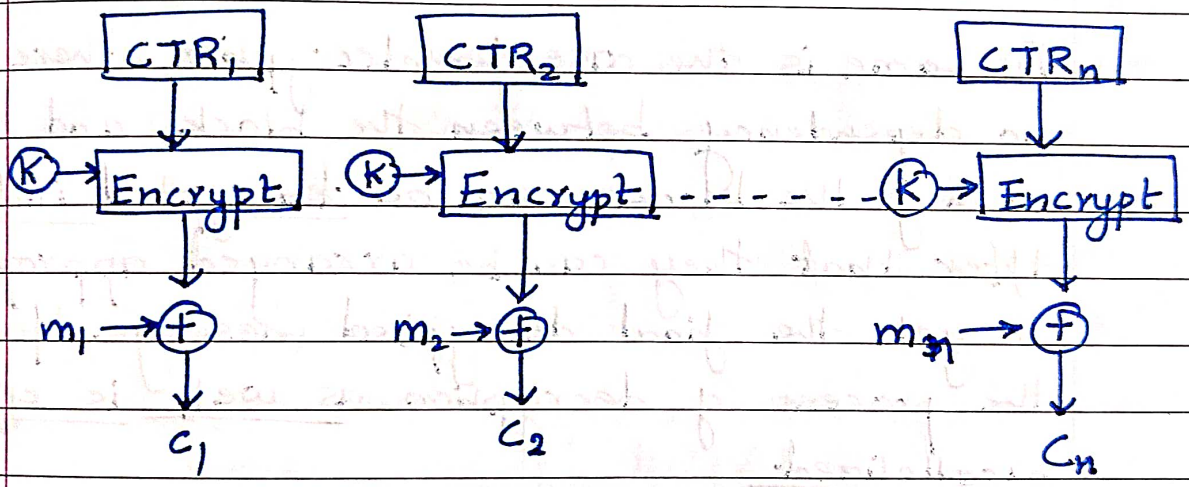
Q.2) In CTR (counter) mode encryption, plaintext is divided into blocks, and each block is encrypted separately using a block cipher (such as AES) with a unique counter value. Decryption in CTR mode is essentially the same process as encryption, just applied in reverse.

$$P = m_1 \,||\, m_2 \,||\, m_3 \,||\, \ldots\ldots \,||\, m_n$$

Enc : $C_i = Enc(CTR_i, K) \oplus m_i$

$i = (0 \text{ to } n)$

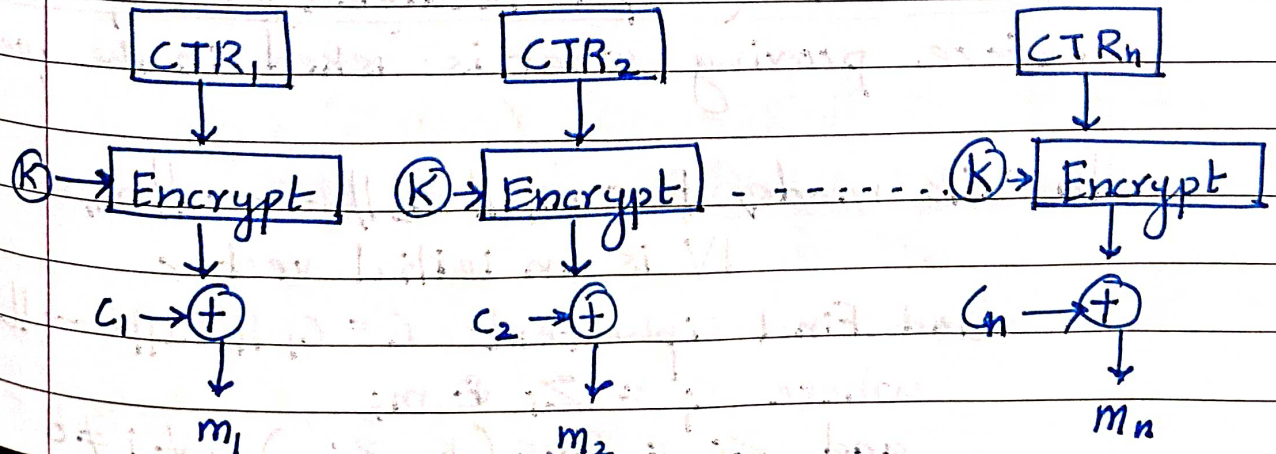
Final ciphertext $C = C_1 \,||\, C_2 \,||\, \ldots\ldots \,||\, C_n$



Dec : $m_i = Enc(CTR_i \oplus K) \oplus C_i$

$i = (0 \text{ to } n)$

Final decryption $M = m_1 \,||\, m_2 \,||\, \ldots\ldots \,||\, m_n$



2021511138
Saidhya Kumar

As we can see from the block diagram of encryption and decryption, each block to be encrypted is independent. There is no dependency between the blocks which are getting encrypted. Each block can be encrypted independently and after all the blocks are encrypted, they can be arranged to get the final ciphertext. There is no chaining as in CBC. Hence, the process is efficiently parallelized.

The same is the case for decryption, there is no dependency between the blocks and hence each of the blocks can be decrypted independently. After that they can be arranged appropriately to get the final decrypted message. Hence, the process of decryption as well is efficiently parallelized.

Q.3) We have already understood CTR mode in the above question. Let us now understand OFB mode (Output Feedback) of operation before proving what is asked in the ques.

In OFB mode, $P = m_1 || m_2 || \cdots || m_n$

IV is an initial vector

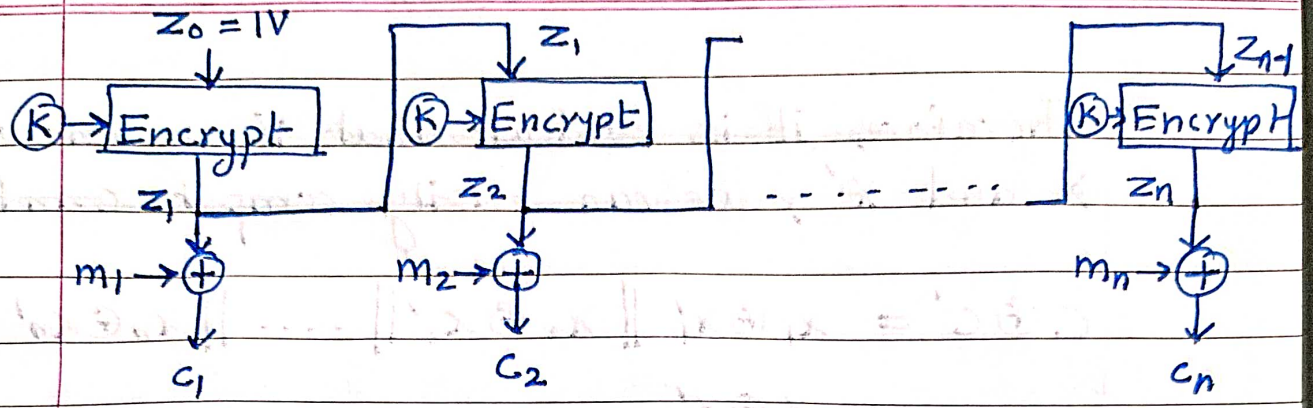
and Final Ciphertext $C = C_1 || C_2 || \cdots || C_n$

where $C_i = Z_i \oplus m_i$

and $Z_i = Enc(K, Z_{i-1}) \quad \forall i \neq 0$

202151138

Sandhya Kumar



Now, as given in the question, we have two sequences of n plaintext blocks $X = (x_1, x_2, \cdots x_n)$ and $X' = (x_1', x_2', \cdots x_n')$. It is also given that both are encrypted using same K and IV

We know that $Z_i = Enc(k, Z_{i-1})\ \forall i \neq 0$ and if K and IV are same then $Z_i = Z_i'$ where $Z_i' = Enc(K, Z_{i-1}')$ because :—

$Z_0 = IV$	$Z_0' = IV$ ($\because$ same)
$Z_1 = Enc(K, IV)$	$Z_1' = Enc(K, IV)$
$Z_2 = Enc(K, Z_1)$	$Z_2' = Enc(K, Z_1')$

Note that $Z_0 = Z_0'$, $Z_1 = Z_1'$, $Z_2 = Z_2'$ and hence $Z_i = Z_i'$ for $\forall i \neq 0$

Now, let consider C_t is ciphertext corresponding to X where $C = (C_1, C_2, C_3 \cdots C_n)$ and C' is ciphertext corresponding to X' where $C' = (C_1', C_2', \cdots C_n')$.

$\therefore C = C_1 \,||\, C_2 \,||\, \cdots \,||\, C_n$

$C = (Z_1 \oplus x_1) \,||\, (Z_2 \oplus x_2) \,||\, \cdots \,||\, (Z_n \oplus x_n)$

and $C' = (Z_1 \oplus x_1') \,||\, (Z_2 \oplus x_2') \,||\, \cdots \,||\, (Z_n \oplus x_n')$

$C \oplus C' = [(Z_1 \oplus x_1) \oplus (Z_1 \oplus x_1')] \,||\, \cdots \,||\, [(Z_n \oplus x_n) \oplus (Z_n \oplus x_n')]$

$C \oplus C' = [x_1 \oplus x_1'] \,||\, [x_2 \oplus x_2'] \,||\, \cdots \,||\, [x_n \oplus x_n']$

2021511038
Saidhya Kumar

Therefore, it is evident that if we know X and X', we can easily compute C and C'

$$C \oplus C' = x_1 \oplus x_1' \,||\, x_2 \oplus x_2' \,||\, \dots \,||\, x_n \oplus x_n'$$
$$= X \oplus X'$$

Therefore, it is evident that $X \oplus X'$ can be computed easily by doing $C \oplus C'$ given C and C' and given X and X' are encrypted using OFB mode of operation using same K and IV. Hence, proved

Similarly, if CTR mode is used —

$$C = C_1 \,||\, C_2 \,||\, \dots \,||\, C_n$$
$$= Enc(CTR_1, K) \oplus x_1 \,||\, \dots \,||\, Enc(CTR_n, K) \oplus x_n$$

and K

$\because CTR_1$ is reused $\therefore CTR_1 = CTR_2 = \dots = CTR_n$

$\therefore Enc(CTR_1, K) = Enc(CTR_2, K) \dots$

lets call it z_i

$$\therefore C = z_1 \oplus x_1 \,||\, z_2 \oplus x_2 \,||\, \dots \,||\, z_n \oplus x_n$$
$$\text{and } C' = z_1 \oplus x_1' \,||\, z_2 \oplus x_2' \,||\, \dots \,||\, z_n \oplus x_n'$$

$$C \oplus C' = [(z_1 \oplus x_1) \oplus (z_1 \oplus x_1')] \,||\, \dots \,||$$

$$C \oplus C' = x_1 \oplus x_1' \,||\, x_2 \oplus x_2' \,||\, \dots \,||\, x_n \oplus x_n'$$
$$= X \oplus X'$$

Therefore, it is evident that $X \oplus X'$ can be computed easily given C and C' by simply

20215138
Savidhya Kumar

doing $C \oplus C'$ ang given that CTR mode is used where CTR and K are reused.

Q.4) (a) $f(x) = x^4 + x + 1$

We are given the connection polynomial, let us find the linear feedback function from this—

We know,

$\therefore L = C_1 S_{n-1} \oplus C_2 S_{n-2} \oplus \cdots C_n S_0$

$f(x) = 1 + C_1 x + C_2 x^2 + \cdots C_n x_n$

for any feedback fn L with n states.

$\therefore f(x) = x^4 + x + 1 \Rightarrow L = C_1 S_{n-1} \oplus C_4 S_0$

$\Rightarrow L = C_1 S_3 \oplus C_4 S_0$

$\Rightarrow L = \boxed{S_3 \oplus S_0}$

Hence, the LFSR will be as follows—

C_1	C_2	C_3	C_4
S_3	S_2	S_1	S_0

$L(S) = S_0 \oplus S_3$

Let us now find the period of this LFSR. Before that we need to check if $f(x) = x^4 + x + 1$ is primitive or not. If we can generate all polynomials of deg < 4 then this $f(x)$ is primitive. (under modulo $f(x)$).

$x^0 = 1$

$x^1 = x$

$x^2 = x^2$

$x^3 = x^3$

Note: use $x^4 = x + 1$ to reduce deg > 4.

2021511138
Sanidhya Kumar

$x^4 = x+1$

$x^5 = x(x+1) = x^2+x$

$x^6 = x^2(x+1) = x^3+x^2$

$x^7 = x(x^3+x^2) = x^4+x^3 = x^3+x+1$

$x^8 = x(x^3+x+1) = x^4+x^2+x = x^2+1$

$x^9 = x(x^2+1) = x^3+x$

$x^{10} = x(x^3+x) = x^4+x^2 = x^2+x+1$

$x^{11} = x(x^2+x+1) = x^3+x^2+x$

$x^{12} = x(x^3+x^2+x) = x^4+x^3+x^2 = x^3+x^2+x+1$

$x^{13} = x(x^3+x^2+x+1) = x^4+x^3+x^2+x = x^3+x^2+1$

$x^{14} = x(x^3+x^2+1) = x^4+x^3+x = x^3+x^2+1$

$x^{15} = x(x^3+x^2+1) = x^4+x^2+x = 1$

$x^{16} = x(1) = x$

Period = 15

Starts repeating

We see that it starts repeating after ##15 iterations and also that all the polynomials less than deg 4 is generated using this $f(x)$. Hence $f(x)$ is a primitive polynomial.
If $f(x)$ is primitive polynomial, period of LFSR is $2^n - 1 = 2^4 - 1 = \boxed{15}$.

(b) $f(x) = x^5 + 1$

We need to construct linear feedback function from this $f(x)$.

$$L = c_1 S_{n-1} \oplus c_2 S_{n-2} \oplus \dots c_n S_0$$

$$f(x) = 1 + c_1 x + c_2 x^2 + \dots c_n x_n$$

for any feedback fn L with n states.

2021511138
Savidhya Kumar

$\therefore f(x) = x^5 + 1 \Rightarrow L = C_n S_0$

$\Rightarrow L = C_0 S_0$

$\Rightarrow L = \boxed{S_0}$.

Hence, the LFSR will be —

c_1	c_2	c_3	c_4	c_5
S_4	S_3	S_2	S_1	S_0

; $L(s) = S_0$

Now, this $f(x) = x^5 + 1$ is reducible as follows

$$x^5 + 1 = (x+1)(x^4 + x^3 + x^2 + x + 1)$$

Now, period of this would be LCM of periods of the these factors. But it is difficult to factorize $x^5 + 1$ without using any calculator. So, we can always find period using below method.

$x^0 = 1$
$x^1 = x$
$x^2 = x^2$
$x^3 = x^3$
$x^4 = x^4$

$\boxed{\text{Period} = 5}$

$x^5 = 1 \rightarrow$ Using $x^5 = 1$ from $x^5 + 1 = f(x)$

Hence, period of this LFSR is 5. This 5 can be obtained either by my method above or by calculating LCM of decomposed fns of $f(x)$.

Q.5) $\lambda : Z_{105} \rightarrow Z_3 \times Z_5 \times Z_7$

$\lambda(x) = (x \bmod 3, x \bmod 5, x \bmod 7)$

We need to find λ^{-1} and hence compute $\lambda^{-1}(2,2,3)$

20215138
Sawidhya Kumar

Let $\lambda(x) = (a_1, a_2, a_3)$

and we are given $\lambda(x) = (x \bmod 3, x \bmod 5, x \bmod 7)$

∴ we can write it as follows —

$$x \equiv a_1 \bmod 3$$
$$x \equiv a_2 \bmod 5$$
$$x \equiv a_3 \bmod 7$$

Using Chinese Remainder Thm we can can construct x such that

[A] $\longrightarrow$ $x = (a_1 M_1 M_1^{-1} + a_2 M_2 M_2^{-1} + a_3 M_3 M_3^{-1}) \bmod M$

and this x will satisfy all the 3 eqns above.

We are asked to find λ^{-1} which will take a_1, a_2, a_3 and give me x. Basically, it will be a fn from $Z_3 \times Z_5 \times Z_7 \longrightarrow Z_{105}$.

In order to find λ^{-1} we need to solve eqn [A]

$$\boxed{M = 3 \times 5 \times 7 = 105}$$

$$M_1 = \frac{105}{3} = 35$$

$$M_2 = \frac{105}{5} = 21$$

$$M_3 = \frac{105}{7} = 15$$

Now, M_1^{-1}, M_2^{-1} and M_3^{-1} will be calculated using extended euclidean algorithm.

$$\frac{M}{M_1} \cdot b_1 \equiv 1 \bmod (M_1)$$

$$35 \cdot \underbrace{b_1}_{M_1^{-1}} \equiv 1 \bmod (3)$$

∴ M_1^{-1} is mul. inv of 35 under mod 3.

2021S1138
Sandhya Kumar

3	35	11
	-33	
2	3	1
	-2	
	1	

Using extended euc. algo.

$1 = 3 - 2 \times 1$

$1 = 3 - (35 - 3 \times 11)$

$1 = 12 \times 3 + 35 \times (-1)$

35^{-1}

-1 under mod $3 = 2$

Now, for m_2^{-1} is mul. inv. of 21 under mod 5.

5	21	4
	-20	
	1	

$1 = 21 - 5 \times 4$

$1 = 1 \times 21 - 5 \times 4$

21^{-1}

Now, for m_3^{-1} is mul. inv. of 15 under mod 7.

7	15	2
	-14	
	1	

$1 = 15 - 7 \times 2$

$1 = 1 \times 15 - 7 \times 2$

15^{-1}

$\therefore m_1^{-1} = 2, \quad m_2^{-1} = 1, \quad m_3^{-1} = 1$

$\therefore$ from eqn [A] we get

$$x = (a_1 70 + a_2 21 + a_3 15) \bmod 105$$

$\therefore \lambda^{-1} : Z_3 \times Z_5 \times Z_7 \longrightarrow Z_{105}$

$$\lambda^{-1}(a_1, a_2, a_3) = (70a_1 + 21a_2 + 15a_3) \bmod 105$$

Now, $\lambda^{-1}(2, 2, 3) = [70(2) + 21(2) + 15(3)] \bmod 105$

$= (140 + 42 + 45) \bmod 105$

$= (227) \bmod 105$

$= 17$

$\therefore \lambda^{-1}(2, 2, 3) = 17$

20215138
Saidhya Kumar

Q.6) Given system of congruences —

$x \equiv 12 \bmod 25$

$x \equiv 9 \bmod 26$

$x \equiv 23 \bmod 27$

We can find x using Chinese Remainder Thm

[A] $\longrightarrow$ $x = (25 M_1 M_1^{-1} + 26 M_2 M_2^{-1} + 27 M_3 M_3^{-1}) \bmod M$

$M = 25 \times 26 \times 27 = 17550$

$M_1 = \frac{M}{m_1} = \frac{17550}{25} = 702$

$M_2 = \frac{M}{m_2} = \frac{17550}{26} = 675$

$M_3 = \frac{M}{m_3} = \frac{17550}{27} = 650$

Also, note that 25, 26, 27 are pairwise co-prime since they have no common-factors and hence, we can easily use CRT. Now, let us proceed to find M_1^{-1}, M_2^{-1} and M_3^{-1} using extended euc. algo —

For M_1^{-1} :-

25	702	28
	−700	
2	25	12
	−24	
	1	

$1 = 25 - 2 \times 12$

$1 = 25 - 12 (702 - 25 \times 28)$

$1 = 337 \times 25 - 12 \times 702$

702^{-1}

-12 under 25 is 13

$\therefore$ We obtained $M_1^{-1} = 13$

2021511138
Savidhya Kumar

For M_2^{-1} :-

26	675	25
	−650	
	25	26
	−25	
	1	

$1 = 26 - 1 \times 25$

$1 = 26 - (675 - 25 \times 26)$

$1 = 26 \times 26 - 1 \times 675$

675^{-1}

-1 under mod 26 is 25

$\therefore M_2^{-1} = 25$

For M_3^{-1} :-

27	650	24
	−648	
	2	27
	−26	
	1	13

$1 = 27 - 2 \times 13$

$1 = 27 - 13 \times (650 - 24 \times 27)$

$1 = 313 \times 27 - 13 \times 650$

650^{-1}

-13 under mod 27 is 14

$\therefore$ We obtained $M_3^{-1} = 14$

Now, lets calculate x using CRT (eqn [A])

$x = [12(702)(13) + 96(675)(25) + 23(650)(14)] \bmod 17550$

$x = (109512 + 151875 + 209300) \bmod 17550$

$x = (470687) \bmod 17550$

$\boxed{x = 14387}$

Q.7) Given :- $n = 18923$, $e = 1261$, $c = 6127$ } To find-plaintext

Now, to decrypt this RSA, $\boxed{x = c^d \bmod n}$ we need to find d. To find d, we need to solve the factorization problem.

202151138
Samidhya Kumar

We will find p and q from n, then find $\phi(n)$ and then use $e \cdot d \equiv 1 \bmod \phi(n)$ to finally find our d.

To solve this factorization problem, in order to find two prime p and q such that $n = pq$, we took help of the C code (attached below for reference). From the code, we obtained $p = 127$ and $q = 149$

Now, $\phi(n) = (p-1)(q-1) = (127-1)(149-1)$
$= 126 \times 148 = 18648$

We know, $e \cdot d \equiv 1 \bmod \phi(n)$ ∴ d is basically multiplicative inverse of e under mod $\phi(n)$. We will use extended Euclidean algorithm for this.

```
1261 | 18648 | 14
  -17654
   994 | 1261 | 1
     - 994
      267 | 994 | 3
        -801
         193 | 267 | 1
           -193
            74 | 193 | 2
             -148
              45 | 74 | 1
                -45
                 29
```

continued...

20215138
Sanidhya Kumar

... continued

$29 \mid 45 \mid 1$
-29
$16 \mid 29 \mid 1$
-16
$13 \mid 16 \mid 1$
-13
$3 \mid 13 \mid 4$
-12
1

Using ~~extended algorithm~~ now,

$1 = 13 - 4 \times 3$
$1 = 13 - 4 \times (16 - 13)$
$1 = 5 \times 13 - 4 \times 16$
$1 = 5 \times (29 - 16) - 4 \times 16$
$1 = 5 \times 29 - 9 \times 16$
$1 = 5 \times 29 - 9 \times (45 - 29)$
$1 = 14 \times 29 - 9 \times 45$
$1 = 14 \times (74 - 45) - 9 \times 45$
$1 = 14 \times 74 - 23 \times 45$
$1 = 14 \times 74 - 23 \times (193 - 2 \times 74)$
$1 = 60 \times 74 - 23 \times 193$
$1 = 60 \times (267 - 193) - 23 \times 193$
$1 = 60 \times 267 - 83 \times 193$
$1 = 60 \times 267 - 83 \times (994 - 3 \times 267)$
$1 = 309 \times 267 - 83 \times 994$
$1 = 309 \times (1261 - 994) - 83 \times 994$
$1 = 309 \times 1261 - 392 \times 994$
$1 = 309 \times 1261 - 392 \times (18648 - 14 \times 1261)$
$1 = 5797 \times 1261 - 392 \times 18648$

20215 1138
Saudhya Kumar

Therefore, mul. inv of $e = 1261$ under mod $\phi(n)$ is $d = 5797$. Now, $x = c^d \bmod n$

$$\therefore \boxed{x = (6127)^{5797} \bmod 18923}$$

To efficiently calculate $(6127)^{5797}$ we will use square-and-multiply algorithm.

5797 in binary is :- $[1011010100101]_2$

In square and multiply algo, whenever we encounter 1, in binary rep. of 5797, we will multiply and whenever we encounter 0, we will square it simply :-

here → $c = 6127$

$n = 18923$

1) $c^1 \cdot c^1 = (6127)^2 \bmod n = 15820$
2) $c^{10} \cdot c^{10} = (15820)^2 \bmod n = 15725$
3) $c^{101} = (15725 \times 6127) \bmod n = 10082$
4) $c^{1010} = (10082)^2 \bmod n = 11291$
5) $c^{1011} = (11291 \times 6127) \bmod n = 16392$
6) $c^{10110} = (16392)^2 \bmod n = 9987$
7) $c^{101100} = (9987)^2 \bmod n = 15959$
8) $c^{101101} = (15959 \times 6127) \bmod n = 5652$
9) $c^{1011010} = (5652)^2 \bmod n = 3080$
10) $c^{10110100} = (3080)^2 \bmod n = 5977$
11) $c^{10110101} = (5977 \times 6127) \bmod n = 5074$
12) $c^{101101010} = (5074)^2 \bmod n = 10196$
13) $c^{1011010100} = (10196)^2 \bmod n = 14377$
14) $c^{10110101000} = (14377)^2 \bmod n = 2200$

20215138

Sandhya Kumar

M T W T F S S

Page No.:

Date:

YOUVA

15) $C^{10110101001} = (2200 \times 6127) \bmod n = 6224$

16) $C^{1011010010} = (6224)^2 \bmod n = 2795$

17) $C^{101101001000} = (2795)^2 \bmod n = 15749$

18) $C^{10110100101} = (15749 \times 6127) \bmod n = 5746$

$\therefore \boxed{x = c^d \bmod n = 5746}$

$\therefore$ Plaintext is 5746.

202151138 | Sanidhya Kumar

Code to find prime factors *p* and *q* such that *n = p*q*. Input is ***n = 18923***.

```c
#include <stdio.h>

void primeFactors(int n){
    // Print the number of 2s that divide n
    while (n % 2 == 0){
        printf("%d ", 2);
        n = n / 2;
    }
    // n must be odd at this point. So we can skip one element
    // (Note: i = i + 2)
    for (int i = 3; i * i <= n; i = i + 2){
        // While i divides n, print i and divide n
        while (n % i == 0){
            printf("%d ", i);
            n = n / i;
        }
    }
    // If n is a prime number greater than 2
    if (n > 2){
        printf("%d ", n);
    }
}

int main(){
    int n;
    printf("\nEnter a number to find its prime factors: ");
    scanf("%d", &n);
    printf("\nPrime factors of %d are: ", n);
    primeFactors(n);
    printf("\n\n");
    return 0;
}
```

```
Enter a number to find its prime factors: 18923

Prime factors of 18923 are: 127 149
```

Hence, ***p = 127*** and ***q = 149***.

Q.8) EL : $y^2 = x^3 + 5x + 3$ over Z_{13}.

We wish to find all possible points on EL.

Here x is b/w [0, 12] and y is b/w [0, 12]

A point (x, y) lies on curve if it satisfies the eqn over mod 13. Basically LHS = RHS

where LHS = $y^2 \bmod 13$

RHS = $x^2 + 5x + 3 \bmod 13$

Let us compute a table to store these LHS/RHS val-

LHS		RHS	
y	$y^2 \bmod 13$	x	$x^2+5x+3 \bmod 13$
0	0	0	3
1	1	1	9
2	4	2	8
3	9	3	6
4	3	4	9
5	12	5	10
6	10	6	2
7	10	7	4
8	12	8	9
9	3	9	10
10	9	10	0
11	4	11	11
12	1	12	10

202151138
Savidhya Kumar

Now, we will consider all possible values of (x, y) where LHS = RHS.

(0, 4), (0, 9), (1, 3), (1, 10), (4, 3), (4, 10), (5, 6), (5, 7), (7, 2), (7, 11), (8, 3), (8, 10), (9, 6), (9, 7), (10, 0), (12, 6), (12, 7).

Therefore, there are 17 such points on the given EL.
(Code also attached for cross-verification)

202151138 | Sanidhya Kumar

Code to find all the possible points on an elliptical curve of the form $y^2 = x^3 + ax + b$ over Z_n. Input is ***a = 5***, ***b = 3***, ***n = 13***. Note that code is attached only for cross-verification.

```c
#include <stdio.h>
#include <stdlib.h>

int main() {
    int a, b, n;
    printf("\nEnter the value of a (the coefficient of x): ");
    scanf("%d", &a);
    printf("Enter the value of b (the constant): ");
    scanf("%d", &b);
    printf("Enter the value of n (the modulo): ");
    scanf("%d", &n);
    printf("\nThe points on EL are: \n\n");

    int num_points = 0; // Initialize a variable to count the number of points

    // Iterate through all possible x values (from 0 to n-1)
    for(int i = 0; i < n; i++) {
        int LHS = (i * i) % n; // Compute LHS of EL
        // Iterate through all possible y values (from 0 to n-1)
        for(int j = 0; j < n; j++) {
            int RHS = ((j * j * j) + (a * j) + b) % n; // Compute RHS of EL
            // If the left-hand side equals the right-hand side, it's a valid
point
            if(RHS == LHS) {
                num_points++; // Increment the number of points found
                printf("x = %d\t\ty = %d\n", j, i);
            }
        }
    }

    printf("\nNumber of points: %d\n\n", num_points); // Print the total
number of points found
    return 0;
}
```

202151138 | Sanidhya Kumar

```
Enter the value of a (the coefficient of x): 5
Enter the value of b (the constant): 3
Enter the value of n (the modulo): 13

The points on EL are:

x = 10          y = 0
x = 7           y = 2
x = 1           y = 3
x = 4           y = 3
x = 8           y = 3
x = 0           y = 4
x = 5           y = 6
x = 9           y = 6
x = 12          y = 6
x = 5           y = 7
x = 9           y = 7
x = 12          y = 7
x = 0           y = 9
x = 1           y = 10
x = 4           y = 10
x = 8           y = 10
x = 7           y = 11

Number of points: 17
```

Hence, total number of points are **17** and all the points are **printed** above. The results obtained from code matches with our calculated result.

Therefore, our calculation is **absolutely correct**!

20215138
Samidhya Kumar

Q.9) If the S-box is replaced by a function π_t that is not a permutation, then this means, in particular that π_t is not surjective. Using this we have to derive that a ciphertext-only attack can be used to determine the key bits in the last round, given a sufficient no. of ciphertexts that all have been encrypted using the same key.

Note that in ciphertext-only attacks we are given with set of ciphertexts which are encrypted using same key K and we are to find K and thus plaintexts. lets proceed to solve the question now.

Now, suppose that $\pi_t^{-1}(z) = \phi$ for some $z \in \{0,1\}^4$. Suppose we are given a set of ciphertexts C, all of which are encrypted using the same unknown key, K. For each $y = y_{(1)} || y_{(2)} || y_{(3)} || y_{(4)} \in t$, and for each i, $1 \le i \le 4$, it must be the case that $\boxed{y_{(i)} \oplus K_{(i)} \neq z.}$

For $1 \le i \le 4$, define $\lambda_i = \{0,1\}^4 \setminus \{z \oplus y_{(i)} : y \in C\}$
Then $K_{(i)} \in \lambda_i$, $1 \le i \le 4$. If $|C|$ is reasonably large, then we expect that $|\lambda_i| = 1$ for $1 \le i \le 4$, and hence the key K can be determined in this way.

And hence, a ciphertext-only attack has been successfully derived. (as per question).

Q.10) Given, a hash fn $h: Z_n \times Z_n \to Z_n$, which is a linear function $h(x,y) = ax + by \bmod n$ where $a, b \in Z_n$ and $n \geq 2$.

Lets say we are given the values —

$$h(x_1, y_1) = z_1$$
$$\text{and } h(x_2, y_2) = z_2$$

Let $r, s \in Z_n$ then we have that

$$= h(rx_1 + sx_2 \bmod n,\ ry_1 + sy_2 \bmod n)$$
$$= a(rx_1 + sx_2) + b(ry_1 + sy_2) \bmod n$$
$$= r(ax_1 + by_1) + s(ax_2 + by_2) \bmod n$$
$$= r\,h(x_1, y_1) + s\,h(x_2, y_2) \bmod n$$
$$= rz_1 + rz_2$$

Therefore, given the value of function h at two points (x_1, y_1) and (x_2, y_2), we know its value at various other points, in this case at $(rx_1 + sx_2 \bmod n,\ ry_1 + sy_2 \bmod n)$. This was done without actually having to evaluate

2021511138
Sanidhya Kumar

h at those points (and also note that we do not even need to know the values of constants a and b in order to apply the above-described technique. Hence, proved!

Q.11) [p → prime number
$a, b \in Z_p$
$f(a,b) : Z_p \to Z_p$ by $f_{(a,b)}(x) = ax + b \bmod p$
$x \neq x' \in Z_p$ such that $\begin{cases} f_{(a,b)}(x) = y \\ f_{(a,b)}(x') = y' \end{cases}$
[GIVEN]

$$(ax + b) \equiv y \bmod p \quad \text{—— (A)}$$
$$(ax' + b) \equiv y' \bmod p \quad \text{—— (B)}$$
$$\text{(A)} - \text{(B)}$$
$$a(x - x') \equiv (y - y') \bmod p$$
$$a \equiv (y - y')(x - x')^{-1} \bmod p$$

We know, $x - x' \neq 0 \because x \neq x' \to$ given

So, for checking if we can find a and b, we need to check if $(x-x')^{-1}$ exists or not.
i.e, if $(x-x')$ is invertible or not under mod p
Inverse of $(x-x') \bmod p$ exists iff $\gcd(x-x', p) = 1$

As p → prime no, $\Rightarrow \gcd(x-x', p) = 1$ always
∴ Inverse of $(x-x') \bmod p$ exists.

So, we can find 'a' and 'b'.

20215138
Samidhya Kumar

M T W T F S S
Page No.:
Date:
YOUVA

For a: $a \equiv (y-y')(x-x')^{-1} \bmod p$ ——— [C]

For b: After solving eqn [C], but value of 'a' obtained in eqn (A) and (B) to find 'b'.

Hence, we can derive a and b.