Lecture - 01

Topic: History of Cryptography

Shifted cipher

Each letter is Shifted by k and sent. Eg- ”A” is written as ”A”+k (Shifted by k letters) and sent. This is easy to decode as only 26 (or 36 (if 0-9 nos are included)) possible k are there and thus its easy to check each possibility.

Rolling by wooden stick

A paper is rolled on to a stick and text is written. If seen normally, the letters would look fully shuffled, but if its rolled in the same way as it was written, it can be decoded.
eg- ”MY NAME IS X” is written like

```
M               M              X
    Y       A       E      S
        N              I
```

Thus its crypted as MMXYAESNI.

Mono-Substitution cipher

We have a table where each letter is mapped to other letters and text is ciphered according to that. Here, we have here 26! ways of mapping and so its very difficult to try different possibilities.
This seems like an optimal solution, but there is a problem. In an average english text, each letter has a specific frequency of repetition.
Say letter ”A” is coded to letter ”K” (randomly). So frequency of letter K would be same as of the letter ”A” in a normal text. So by this way, cipher text could possibly be decrypted.

Lecture - 02

Topic: History of Cryptography(Continuation.)

Homophonic Cipher

The main problem of Mono-Substitution cipher is that, a character was substituted with only one alphabet and so the frequency didn't change.
What if its substituted with many characters to equalize the frequencies?
Say $S = \{A, B, ..Z, 0, 1, ..9, \epsilon, \alpha, \beta, \gamma, ...\}$ has usable symbols.
Say letter "A" has frequency $x\%$. We allot $\frac{x}{100} \times |S|$ number of symbols and are randomly substituted in the cipher text in place of "A". This uniforms/balences the frequency among all the symbols and hence difficult to decrypt by frequency method.
But here, storing the mapping, encrypting, and decrypting are difficult.

Vigenere's Cipher

What if we substitute "A" by any of the letters strategically? Vigenere created a table as shown below.

	A	B	C	D	..
A	B	C	D	E	..
B	C	D	E	F	..
.					
.					

A keyword is chosen and correspondingly added to the text encrypt it. Eg - Thus here, according

Actual text	M	Y	N	A	M	E	I	S	X
keyword	R	O	S	E	R	O	S	E	R
Cipher	..			F					

to the position, same letter is encrypted to different letters and thus the frequencies are balenced.
Is it a good method then?
Words like "THE", "IS", etc repeat so much in english that its very likely that it is encrypted to the same cipher text due to same relative position w.r.t keyword. Calculating the repeated strings in ciphertext and observing the distance between them will give insigts about the length of keyword. Length of keyword would be a factor of those distances and can be found out(say l). Now, characters $1, 1+l, 1+2l, ..$ are derived from same column of the table. Hence they are like monostituted and now, frequencies can be calculated out to find the keyletters and hence keyword.

Mordern Cryptography

Shannon's Cipher

$\xi = (E, D)$ is a cipher system where $E(m, k) = c(m$ is message, k is key, c is cipher text) is encyption funtion, and $D(c, k) = m$ is decryption funtion.

One Time Pad

Say m^l is a message of bits of length l, and key k^l is key of same length generated randomly.

$$E(m,k) = m^l \oplus k^l = c$$

$$\begin{aligned} D(c,k) &= c^l \oplus k^l \\ &= m^l \oplus k^l \oplus k^l \\ &= m^l \end{aligned}$$

Provided key is generated completely random, and no part of key is known to Evasdropper, they can't decrypt it as probability of c being 0 or 1 is independent of message itself. I.e,

$$Prob(cipher = c|msg = m) = Prob(cipher = c|msg = m')$$

Hence, it is safe. Disadvantages:

- key is as big as message(or more)
- key should be sent safely. Otherwise its easily decrypted.

If key length is more, either its padded at the end and xored, or key is taken till the length of message and xored.
In general, if its not a bit string, the encryption can be taken as sum modulus like:

$$E(m,k) = m^l + k^l \pmod{n} = c \text{ (if n=2, its just xor)}$$

$$\begin{aligned} D(c,k) &= c^l - k^l \pmod{n} \\ &= m^l + k^l - k^l \pmod{n} \\ &= m^l \pmod{n} \end{aligned}$$

Lecture - 03

Topic: Perfect Secrecy and Shannon's information Theory

Perfectly secrecy

OTP

For a message to be perfectly secret, the Evasdropper should not be able to get any extra information from the ciphertext. So,

$$\begin{aligned}
P(M=m|C=c) &= P(M=m) \quad \text{[message = m, and ciphertext = c]} \\
P_c(m) &= P(m) \\
\frac{P(M=m|C=c)}{P(M=m)} &= \frac{P(C=c|M=m)}{P(C=c)} \\
&= \frac{P(C=c|M=m)}{\sum_{m'\in M} P(C=c|M=m')P(M=m')}
\end{aligned}$$

$$\left[\begin{aligned}
P(C=c|M=m') &= P(K\oplus m'=c|M=m') \\
&= P(K=c\oplus m'|M=m') \\
&= \frac{1}{2^l} \quad \text{[as key is selected randomly, probability that its } c\oplus m' \text{ is } 1/2^l\text{]}
\end{aligned}\right]$$

$$\begin{aligned}
\frac{P(M=m|C=c)}{P(M=m)} &= \frac{P(C=c|M=m)}{\sum_{m'\in M} P(C=c|M=m')P(M=m')} \\
&= \frac{1/2^l}{\sum_{m'\in M}(1/2^l)P(M=m')} \\
&= \frac{1}{\sum_{m'\in M} P(M=m')} \\
&= \frac{1}{1} \\
P(M=m|C=c) &= P(M=m)
\end{aligned}$$

Hence proved that it is perfectly secret.
But, what happens if key is repeated? Say a message said "Fire the gun" to a soldier which was ciphered to c using key k, though an Evasdropper technically doesn't know the key, now he would see the soldier firing after getting message and so he can guess the message. Using the ciphertext, he can get the $key = message \oplus cipher$ and if same key is used again, he would guess the message. Thus key can be used just once.
Also, if $M = m_1 =' a', m_2 =' ab'$ and,
if $c =' x'$, $P_c(m_1) = 1$ and $P_c(m_2) = 0$ (This method reveals length of the message)
if $c =' xy'$, $P_c(m_1) = 0$ and $P_c(m_2) = 1$.

Substitution cipher

If $M = m_1 =' aa', m_2 =' ab'$ and,
if $c =' xx'$, $P_c(m_1) = 1$ and $P_c(m_2) = 0$.
if $c =' xy'$, $P_c(m_1) = 0$ and $P_c(m_2) = 1$.
Thus its not perfectly secret.

Addition OTP

$$\begin{aligned} D(c,k) &= c^l - k^l \pmod n \\ &= m^l + k^l - k^l \pmod n \\ &= m^l \pmod n \end{aligned}$$

Proof is very similar to as OTP.

Shannon's information Theory

"No class on Friday" has more information/importance than "There is class on Friday" because having no class is a rare thing, and need to informed importantly. Having class is a regular thing and it doesn't carry much info. So,

$$\begin{aligned} \text{information} &\propto \frac{1}{\text{probability of occurance}} \\ Info(x) &\propto \frac{1}{P(x)} \end{aligned}$$

Entropy of a message distibution(X) is defined as:

$$\begin{aligned} H(X) &= -\sum_{x \in X} P(x) \log_2(P(x)) \\ &= \sum_{x \in X} P(x) \log_2(\frac{1}{P(x)}) \end{aligned}$$

Entropy is max when each of the messages has equal probability i.e, they are more uncertain.
Conditonal entropy of X, given Y is:

$$\begin{aligned} H_Y(X) &= \sum_{X,Y} P(x,y) \log_2(\frac{1}{P_y(x)}) \\ &= \sum_{Y} P(y) \sum_{X} P(x) \log_2(\frac{1}{P_y(x)}) \end{aligned}$$

If C is the cipher text, and if $H_C(M) \approx 0$, then its easily breakable as it is not that uncertain.

Lecture - 04

Topic: Key distributions

Symmetric Key Cryptography

The methods we have seen so far including Substitution cipher, OTP, etc, are Symmetric key Cryptography as both the sender and receiver needs the same key to encrypt and decrypt the message. Here, the main difficulty was to exchage keys between both parties safely.
Its two types are **Stream Cipher** and **Block cipher** which we'll see later.

Assymetric key Cryptography

Here, a pair of Keys E_k and D_k are created by a party which are related to each other in some sense.

Method 1: Secrecy Ensured

Lets say, person A wants to send a message to person B. Now, person B created the pair of keys E_B and D_B and sends E_B publically. So, now A encrypts message m using E_B to send cipher c.
Here, since D_B is known only by person B, just B can decrypt it and no one else. Thus here, secracy is ensured. But, cipher c can be tapped and some other message m' can be encrypted to c' using public key E_B by an Evasdropper and sent. So, here, authenticity is not ensured.

$$\mathrm{m} \longrightarrow \underbrace{\boxed{E_B}}_{E_B(m)} \longrightarrow \mathrm{c} \qquad \mathrm{c} \longrightarrow \underbrace{\boxed{D_B}}_{D_B(E_B(m)) = m} \longrightarrow \mathrm{m}$$

Method 2: Authenticity Ensured

Lets say, person A wants to send a message to person B. Now, person A created the pair of keys E_A and D_A and sends E_A publically. So, now A encrypts message m using D_A to send cipher c.
S Here, since E_A is known by everyone including B, he can decrypt it using E_A. Thus here, authenticity is ensured as D_A is private to person A and only he/she can encrypt it. But, cipher c can be decrypted by littrally everyone as E_A is known publically. So security is not ensured.

$$\mathrm{m} \longrightarrow \underbrace{\boxed{D_A}}_{D_A(m)} \longrightarrow \mathrm{c} \qquad \mathrm{c} \longrightarrow \underbrace{\boxed{E_A}}_{E_A(D_A(m)) = m} \longrightarrow \mathrm{m}$$

Method 3: Both Ensured

What if we combine both the above methods to ensure both..
Lets say, person A wants to send a message to person B.
Both of them creates pairs of keys E_A, D_A and E_B, D_B (and, E_B, E_A are public).

$$m \longrightarrow \boxed{D_A} \longrightarrow \boxed{E_B} \longrightarrow c \qquad c \longrightarrow \boxed{D_B} \longrightarrow \boxed{E_A} \longrightarrow m$$

$$D_A(m) \qquad E_B(D_A(m)) \qquad\qquad D_B(E_B(D_A(m))) \quad E_A(D_B(E_B(D_A(m))))$$

$$= D_A(m) \qquad\qquad\qquad = m$$

This suffices both authenticity and secracy. Here, its noteworthy that algorithm is known by everyone unlike the historical methods and just that the keys are kept secret.

Access Control

Eg- MAC, DAC, RBAC, ABAC, etc are algorithms/methods used to enable access control. Lets understand this with an eg:
Lets say there is data stored in a database where only specific users can read and special users can edit it. Also, people should not be able to delete or scatter the information. So, readers and modifiers should have their own specific keys.

Random Number Generation

Its of two types:

True Random Number generator(TRNG)

This is based on actual random events such as some hardware that changes drastically with outside conditions. It is truely random and can't be predicted.

Pseudo Random Number generator(PRNG)

This is done algorithmic and could be predicted based on its previous values.

Lecture - 05

Topic: Symmetric Crytopgraphy(Stream Cipher, LFSR)

Weekly Test 1 Solutions

Question 2

Given :
Length of code = 128 bits
Cost of a processor = Rs.1000
Cap on cost of processors is Rs.10 crores. The performance of the processors is 10ns/code and follows Moore's law, i.e. it doubles in 24 months. The code is expected to be broken in 7 days.

Solution:
In n years, performance will increase by a factor of $2^{\frac{n}{2}}$. Therefore,

$$2^{128} codes \times \frac{10 \times 10^{-9} s}{2^{\frac{n}{2}}} = \frac{10 crore}{1000} processors \times 7 days \times 24 hrs \times 60 min \times 60 s$$

On solving, n turns out to be approximately 130 years.

Symmetric Encryption: Stream Cipher

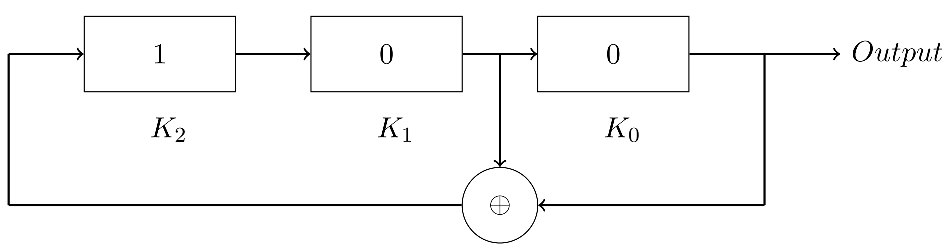
The input is a stream of bits and the encryption takes place one bit at a time. It is easy to compute. e.g.: One Time Pad which encrypts as follows

$$c_i = m_i \oplus k_i$$
$$c_i = m_i + k_i \bmod 2$$

To use this encryption we need a random number generator to obtain k. True random numbers can be generated from physical phenomena. Rather, We are gonna generate pseudo-random numbers, i.e., random numbers generated algorithmically. One such method is LFSR.

Linear Feedback Shift Register

LFSR is a shift register whose input bit is a **linear** function of two or more of the previous output bits.



The sequence generated by the given circuit is

$$K_{i+3} = K_{i+1} \oplus K_i$$
$$K_{i+3} = K_{i+1} + K_i \bmod 2$$

The expression for the input bit of an LFSR can be represented by a polynomial of degree n (n = number of registers), known as the characteristic polynomial. For example, the polynomial of the above LFSR circuit is

$$x^3 = x + 1 \bmod 2$$
$$x^3 + x + 1 \bmod 2 = 0$$

Such a polynomial is called primitive if it is a factor of $x^{2^n-1} + 1 \bmod 2$. A primitve polynomial generates a maximum length cycle of register values for given number of registers (cycle length $= 2^n - 1$, every pattern except 0). For example, the above polynomial generates the following series:

$$100, 010, 101, 110, 111, 011, 001$$

of length $2^3 - 1 = 7$. Note that the polynomial is a factor of $x^7 + 1 \bmod 2$.

$$x^{2^n-1} + 1 \bmod 2 = (x+1) \times (x^3 + x + 1) \times (x^3 + x^2 + 1) \bmod 2$$

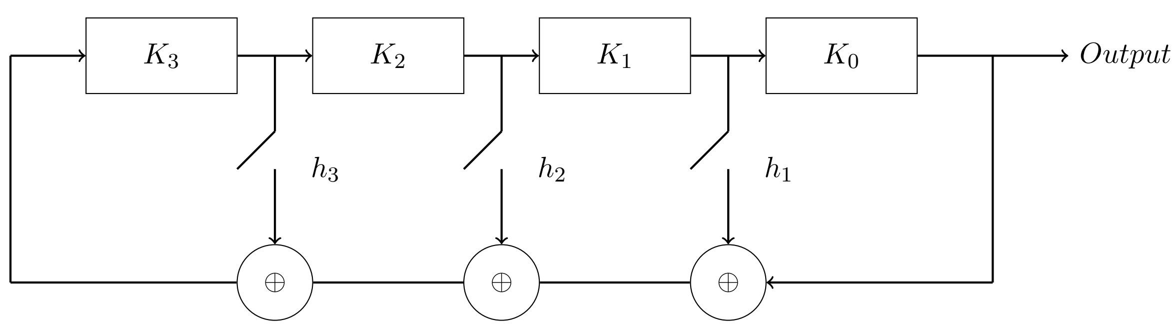
To preserve randomness, the length of the cycle should be maximized, and so a primitive polynomial is preferred. Otherwise, we will end up generating a cyle of length less than $2^n - 1$.

Here, the key depends on

- Characteristic polynomial
- **Seed** (Initial Value of the registers)

Programmable LFSR

An LFSR circuit of degree n that can configured to adopt any characteristic polynomial of the same degree. Here's an example of a programmable LFSR of degree 4.



The sequence generated by this would depend on the keys h1,h2,h3 as shown:

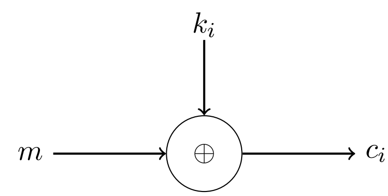
$$K_{i+4} = h_3 K_{i+3} + h_2 K_{i+2} + h_1 K_{i+1} + K_i \bmod 2$$

Due to its linear nature, an LFSR can be easily broken. We shall introduce non-linearity by using AND and OR gates in the circuit which will be covered in the next class :)

Lecture - 06

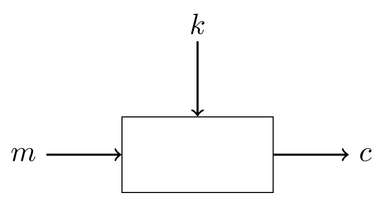
Topic: DES, Feistel Cipher

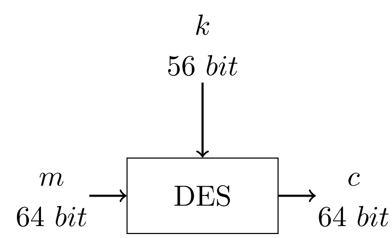
Stream Cipher - Short Summary



Block Cipher

Bundle of bits are fed.





Principle

$$m \longrightarrow \text{confusion} \longrightarrow \text{diffusion} \longrightarrow c$$

The text undergoes several iterations of confusion and diffusion.

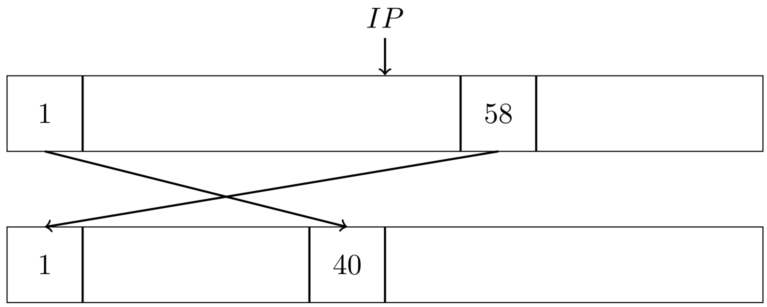
Confusion

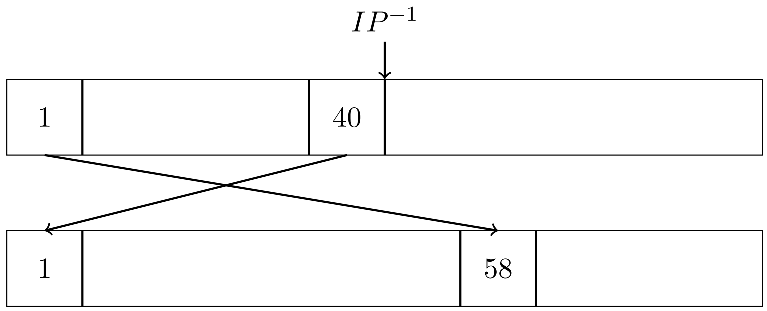
Relation between the text message and ciphertext is obscure.

Diffusion

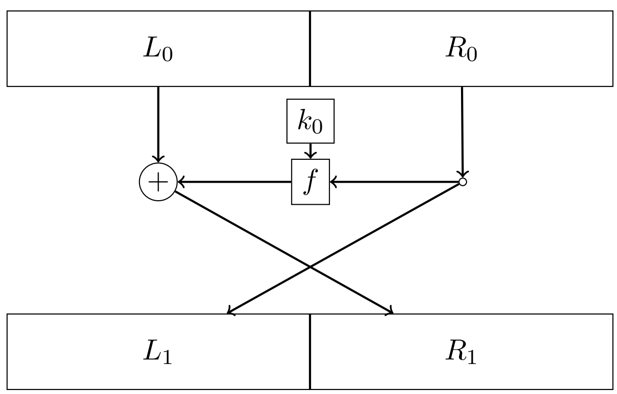
Change in one bit in the plaintext influences multiple bits in the ciphertext.

Confusion + Diffusion = Security ;)



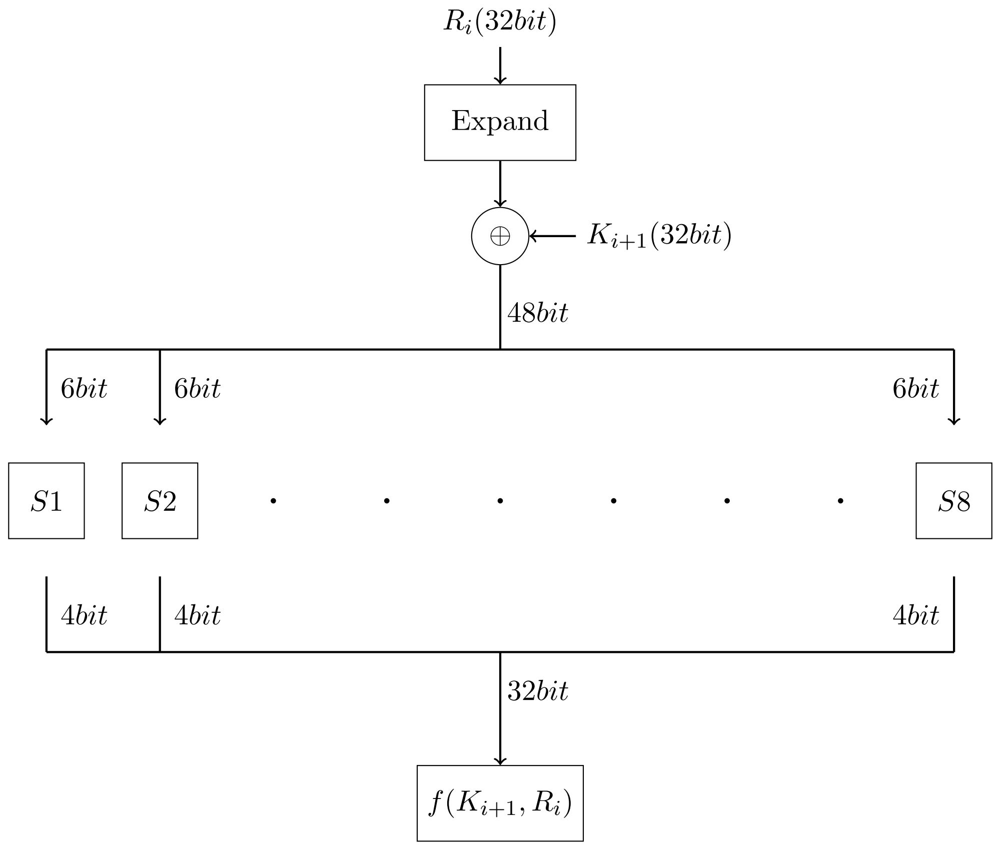


Feistel Cipher



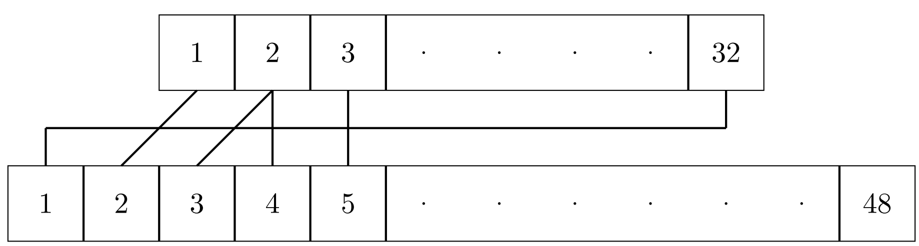
- Left half(L0) is encrypted in one round and put on the right side(R1).
- Right half(R0) is simply copied to the left side(L1).
- The function f depends on the right half (here, R0) and the corresponding key of that round.

Function $f(K_{i+1}, R_i)$



Expansion

1 bit may expand to 1 or 2 bits.

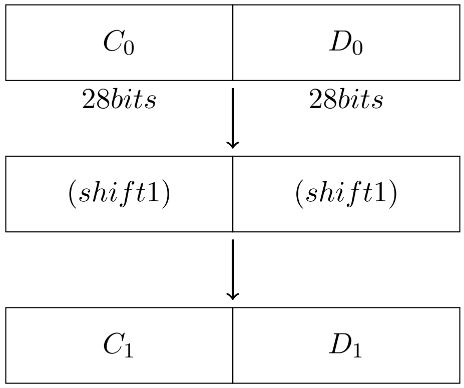


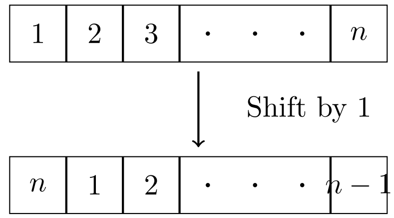
Substitution

Substitution is done based on 8 tables(S1 to S8) of size 4×16. The 6 bit input is used to navigate through the substitution table. The first and last bit indicates the row and the 4 bits in the middle indicate the column.

Key Generation

The Key is of 64 bits, outta which every 8^{th} bit is redundant and thus discarded. Let us look at the 56 bits that matter,





Note : The left half and right half of first key are C_0 and D_0, respectively. Similarly, the second key comprises of C_1 and D_1.

- The keys for each round are generated by shifting from the preceding key.
- In rounds 1, 2, 9, and 16, shift occurs with an offset of 1, while for the remaining rounds, the offset is 2.

Lecture - 07

Topic: Decryption of DES and Assymetric Cryptography

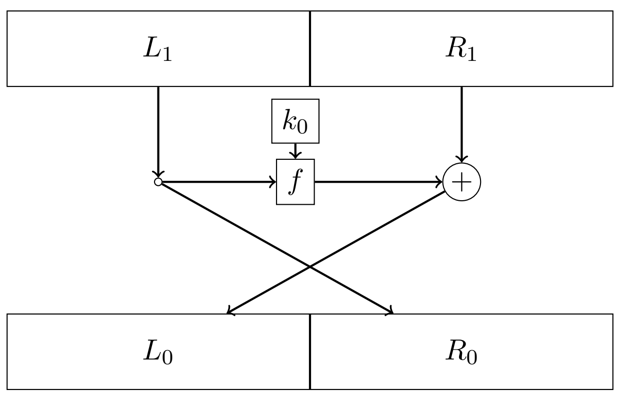
Last class we saw about encryption in DES. Now lets see how to decrypt it.

Decryption in DES

For decryption we just follow the exact same enryption method in reverse order. You may think that we need to calculate f^{-1} which had expansion and contraction using s-boxes. But actually, we don't need its inverse and we can use the same f function.

$$\begin{aligned} R_{i+1} &= L_i \oplus f(k_i, R_i) \\ L_i \oplus R_{i+1} &= L_i \oplus L_i \oplus f(k_i, R_i) \\ L_i \oplus R_{i+1} &= f(k_i, R_i)\ [R_i = L_{i+1}] \\ L_i \oplus R_{i+1} &= f(k_i, L_{i+1})\ [R_i = L_{i+1}] \\ L_i &= R_{i+1} \oplus f(k_i, L_{i+1}) \end{aligned}$$

Thus, we get L_i from R_{i+1} and L_{i+1}.



As we have done the reverse, for 1 round, if we do it for 16 rounds, the final ouptut comes in R_0 and L_0 in reverse order and thus it should be reversed to get the original message.

Lets prove that its in reverse order. [d- denotes for bitstrings during decryption, and e- denotes encryption]

$$
\begin{aligned}
L_0^d &= R_{16}^e \\
R_0^d &= L_{16}^e = R_{15}^e \\
L_1^d &= R_0^d = R_{15}^e \\
R_1^d &= L_0^d \oplus f(R_0^d, k_1^d) \\
&= L_0^d \oplus f(R_1 5^e, k_1 6^e) \\
&= R_1 6^e \oplus f(R_1 5^e, k_1 6^e) \\
&= L_1 5^e \oplus f(R_1 5^e, k_1 6^e) \oplus f(R_1 5^e, k_1 6^e) \\
&= L_1 5^e
\end{aligned}
$$

thus, $R_1^d = L_1 5^e$ and similarly, $l_1^d = R_1 5^e$. so just the right and left are just reversed.

Problem with DES, and subsequent methods

Here, since the key length is just 64 bits, brute force method was very much possible, and so other methods such as AES, DES were introduced.

AES(Advanced Encryption Standard)

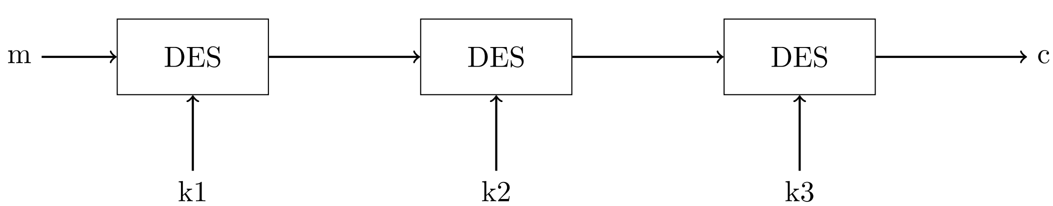
AES is a block cipher which encrypts 128 bits at a time. Its key length are 128 bits, 192, or 256 in different different methods. Correspondigly, 10 rounds, 12, and 14 rounds of encryption happens respectively.
Unlike DES, here every operation is byte(8 bits) wise and not bit wise.
In every round, the following happens:

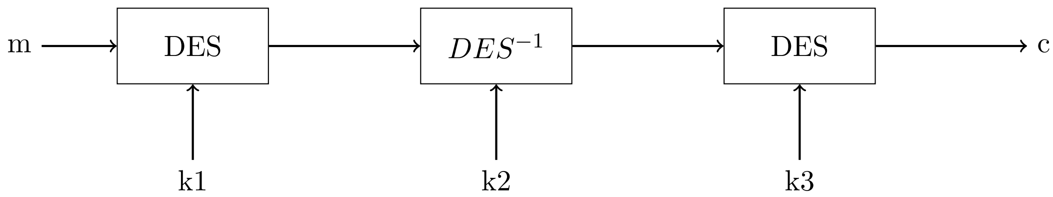
- byte Substitution(by passing thorough s-blocks and stuffs)
- Row shift (for diffusion)
- Mix column (for diffusion)
- Key addition(xor with key)

Since they are byte wise operation, (i.e, 0,1,2,..,255) finite field algebra would be useful to understand AES. Galois field theory (GF(256)) - prof suggested to read this on our own.

3DES or TDES(triple DES)

It can be done is two ways as shown:





Rivest–Shamir–Adleman (RSA) algorithm

Its an Assymetric type encryption. The method is as follows.

- get two large prime nos p and q (let $p, q > 2^{1024}$)
- find $n = pq$
- euler's function $\phi(n) = (p-1)(q-1)$
- select public key e such that, $gcd(e, \phi(n) = 1)$
- obtain private key d which is $= e^{-1} \pmod{\phi(n)}$. Here, inverse element exist because $gcd(e, \phi(n) = 1)$.

private info - d,p,q
public info - e,N

encrypt: $c = m^e \pmod{n}$

decrypt:

$$\begin{aligned} c^d \pmod{n} &= (m^e)^d \pmod{n} \\ &= m^{ed} \pmod{n} \\ &= m^{q\phi(n)+1} \pmod{n} \\ &= m^{q\phi(n)}.m \pmod{n} \\ &= m \text{ ;fermat's little theorem} \end{aligned}$$

fermat's little theorem - $a^{\phi(n)} \pmod{n} = a$.

Generally, Assymetric key is computationally very expensive and thus very difficult to send the entire message through Assymetric method. Thus, keys of Symmetric key method are sent using Assymetric method and messages are sent with Symmetric key encryption henceforth.

Lecture - 08

Topic: RSA algorithm and Diffie–Hellman Key Exchange

Breaking RSA

The main difficulty of breaking RSA algorithm lies in factorisation of n. As seen in previous lecture, if we could find p and q from n, then $\phi(n)$ can be found out. Using e (Public key), d can be found and thus could be decrypted by any Eavesdropper. Thus factorisation is the only difficulty here.. In fact, such a factorisation is an open problem, if factorised, awards are awaiting.. See link

Diffie–Hellman Key Exchange

Decide on prime number p and a primive root α(Primive root will be explained later. For now, assume its a number from 1 to $p-1$). Both these are public info.

A chooses a private key a from 2,3,..,$p-2$ (Later would be explained why 1 and $p-1$ are not included.) Calculate public key $A = \alpha^a \pmod p$ Send A Recieve B. $K_{AB} = B^a \pmod p = \alpha^{ab} \pmod p$	A chooses a private key b from 2,3,..,$p-2$ (Later would be explained why 1 and $p-1$ are not included.) Calculate public key $B = \alpha^b \pmod p$ Send B Recieve A. $K_{AB} = A^a \pmod p = \alpha^{ab} \pmod p$

Now, using K_{AB}, Symmetric Cryptography techniques can be applied to send long messages.

Breaking DHP(Diffie–Hellman Problem)

Eavesdropper knows α, p, A, B. The only unknown things are a, b.

$$A = \alpha^a \pmod p$$
$$a = log_{\alpha}^{A} \pmod p$$

Solving this Discrete Logarithm Problem(DLP) is computationally very hard.

Group Theory

A group is a set of elements and an operator * such that it follows the following properties:

- Closure property: if $a, b \in G, a * b = c \in G$
- Associativity
- existance of neutral element(identity element)
- existance of inverse element

(Commutativity is not necessary but if there, its called an abelian group)

How to choose α

Here, lets define a group $\mathbb{Z}_p^* = i : i \in 0, 1, .., p-1, gcd(i, p) = 1$ with operator as multiplication in $\pmod{p}$.
If p is a prime number, then $\mathbb{Z}_p^* = i : i \in 1, .., p-1, gcd(i, p) = 1$
Euler's function, $\phi(p) = |\mathbb{Z}_p^* = p - 1|$ for a prime number.
Fermat's little theorem states that, $a^{\phi(n)} \pmod{n} = a$ for any n. So here, $\alpha^{\phi(p)} \pmod{p} = \alpha^{p-1} \pmod{p} = 1$
i.e, on powering α repeats in a cycle of length, a factor of $\phi(p) = p - 1$. If the length is exactly $p - 1$, we say its a primitive root of p. We want the length to be as long as possible to maximise randomness, and number of possiblities of α^n.
Eg - in $\mathbb{Z}_{11}^*$, 2 is a primitive root whereas, 3 is not.. 3 has a cycle of 5, and 2 has a cycle of 10 (= 11-1). Try it out.

Lecture - 09

Topic: Diffie-Hellman Key Exchange Protocol (DHD), Elgamal

Diffie-Hellman Key Exchange

Say there are two parties A and B that want to communicate:
Public info : p, α

Party A
$K_{private(A)} = a \in 2, 3, ..., p-2$
$K_{public(A)} = A = \alpha^a \pmod{p}$

Party B
$K_{private(B)} = b \in 2, 3, ..., p-2$
$K_{public(B)} = B = \alpha^b \pmod{p}$

$$
\begin{aligned}
& \xrightarrow{\text{A}} K_{AB} = A^b \pmod{p} \\
& \qquad = \alpha^{ab} \pmod{p} \\
K_{AB} = B^a \pmod{p} & \xleftarrow{\text{B}} \\
= \alpha^{ab} \pmod{p} & \\
c = m.K_{AB} \pmod{p} & \xrightarrow{\text{c}} m = c.K_{AB} \pmod{p}
\end{aligned}
$$

Elgamal(1985)