CS 745 : Principles of Data and System Security

Scribed by: Kavin Arvind and Nikil S

Contents

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 Eavesdropper, 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 Eavesdropper should not be able to get any extra information from the ciphertext. So,

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

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

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

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 Eavesdropper 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 Eavesdropper and sent. So, here, authenticity is not ensured.

$$\text{m} \longrightarrow \boxed{E_B} \longrightarrow \text{c} \qquad \text{c} \longrightarrow \boxed{D_B} \longrightarrow \text{m}$$

$$E_B(m) \qquad\qquad D_B(E_B(m)) = m$$

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.

$$\text{m} \longrightarrow \boxed{D_A} \longrightarrow \text{c} \qquad \text{c} \longrightarrow \boxed{E_A} \longrightarrow \text{m}$$

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

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).

$$m \longrightarrow \boxed{D_A} \longrightarrow \boxed{E_B} \longrightarrow c$$

$$D_A(m) \qquad E_B(D_A(m))$$

$$c \longrightarrow \boxed{D_B} \longrightarrow \boxed{E_A} \longrightarrow m$$

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

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

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

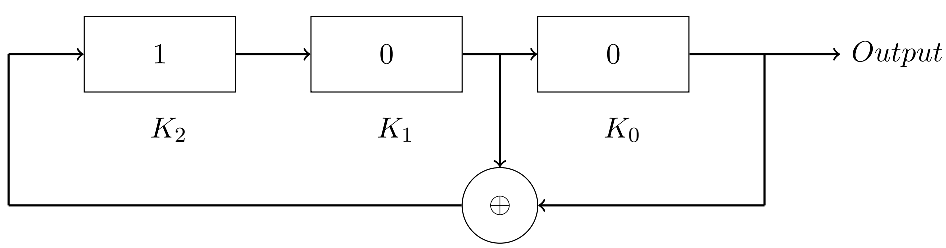
The input is a stream of bits and the encryption takes place one bit at a time. It is easy to compute. e.g.: One Time Pad which encrypts as follows

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

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 $x^7 + 1 \bmod 2$.

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

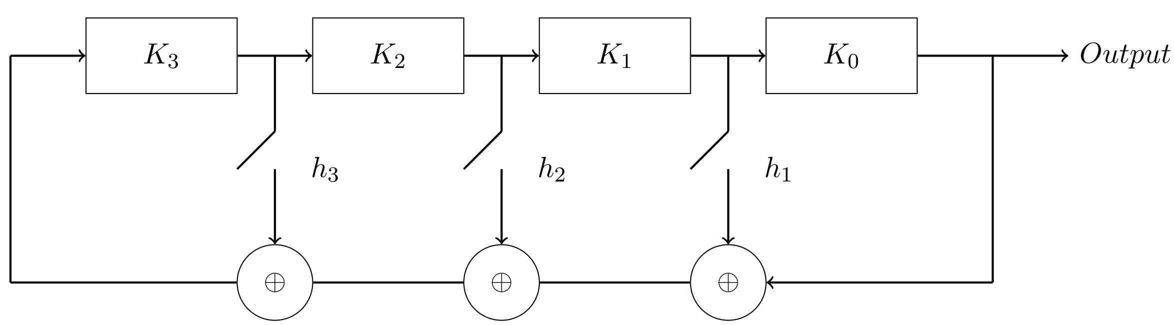
To preserve randomness, the length of the cycle should be maximized, and so a primitive polynomial is preferred. Otherwise, we will end up generating a cyle of length less than $2^n - 1$.

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:

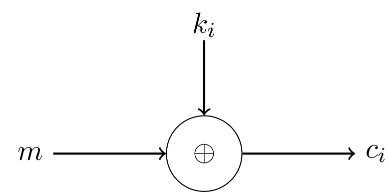
$$K_{i+4} = h_3 K_{i+3} + h_2 K_{i+2} + h_1 K_{i+1} + K_i \bmod 2$$

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

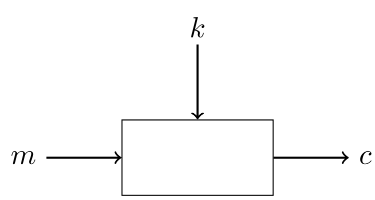
Topic: DES, Feistel Cipher

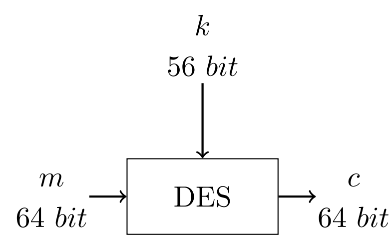
Stream Cipher - Short Summary



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.

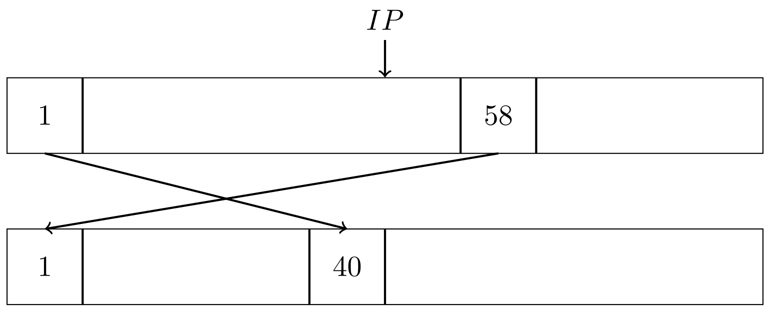
Confusion

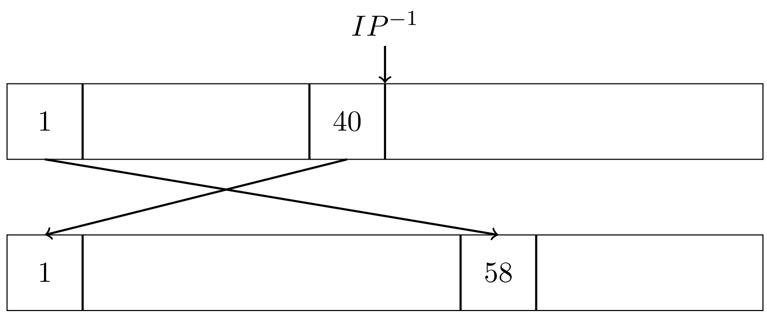
Relation between the text message and ciphertext is obscure.

Diffusion

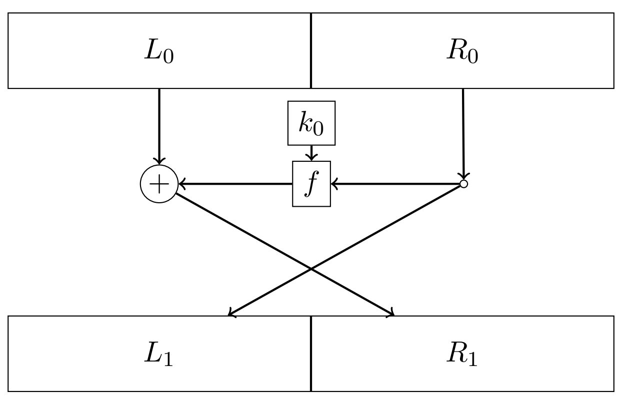
Change in one bit in the plaintext influences multiple bits in the ciphertext.

$$\boxed{\text{Confusion} + \text{Diffusion} = \text{Security ;)}}$$



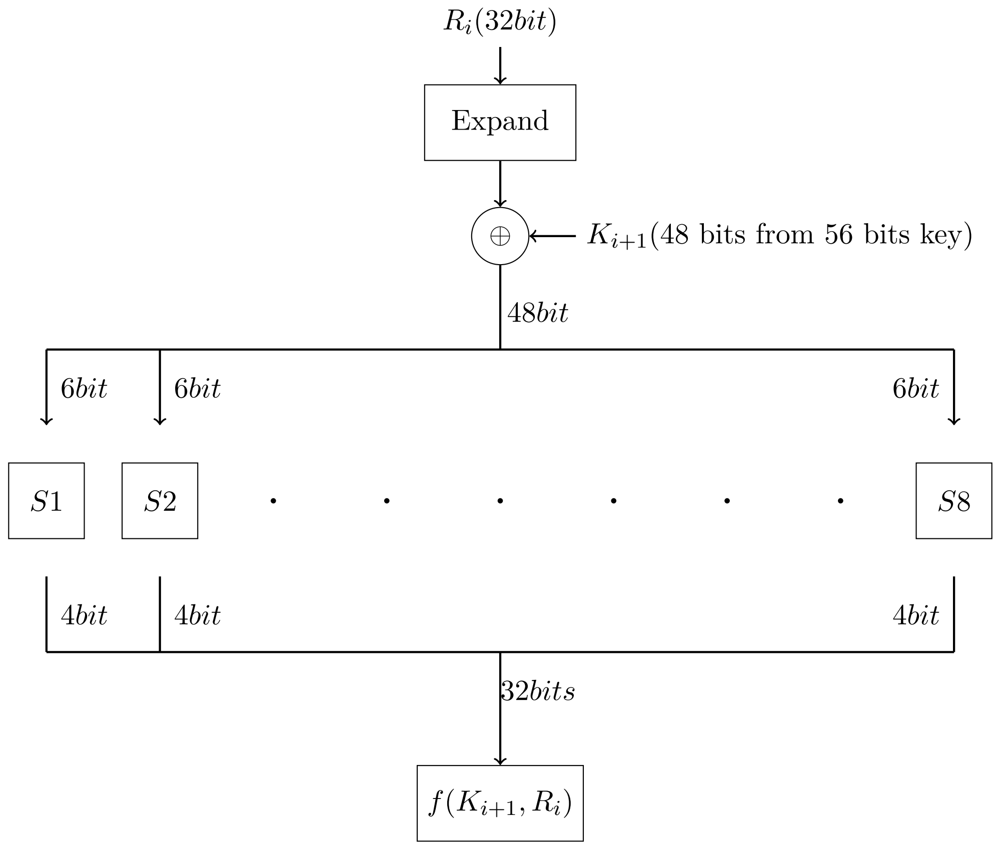


Feistel Cipher



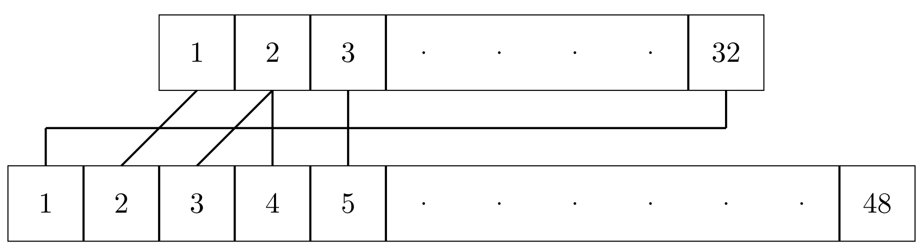
- Left half(L0) is encrypted in one round and put on the right side(R1).
- Right half(R0) is simply copied to the left side(L1).
- The function f depends on the right half (here, R0) and the corresponding key of that round.

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



Expansion

1 bit may expand to 1 or 2 bits.

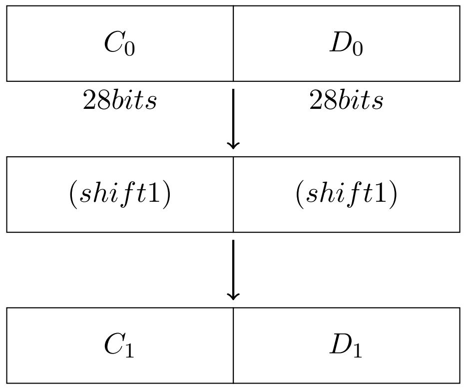


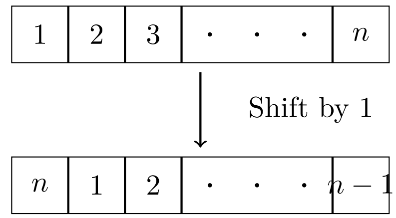
Substitution

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 56 bits, outta which every 7^{th} bit is redundant and thus discarded after shifting(to be precise, its rotation) as shown below.





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.

Thus after rotating, we discard the 7th bits to get 48 bit keys for every corresponding round.

- 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

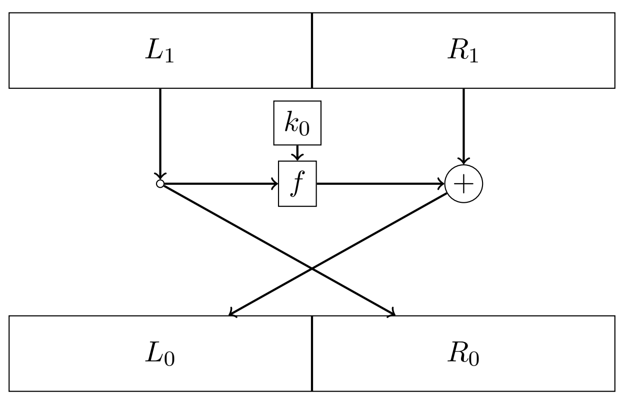
Last class we saw about encryption in DES. Now lets see how to decrypt it.

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]

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

thus, $R_1^d = L_{15}^e$ and similarly, $L_1^d = R_{15}^e$. 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)

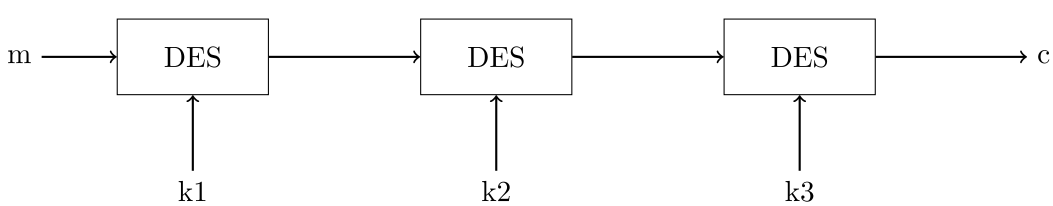
AES is a block cipher which encrypts 128 bits at a time. Its key length are 128 bits, 192, or 256 in different different methods. Correspondigly, 10 rounds, 12, and 14 rounds of encryption happens respectively.
Unlike DES, here every operation is byte(8 bits) wise and not bit wise.
In every round, the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

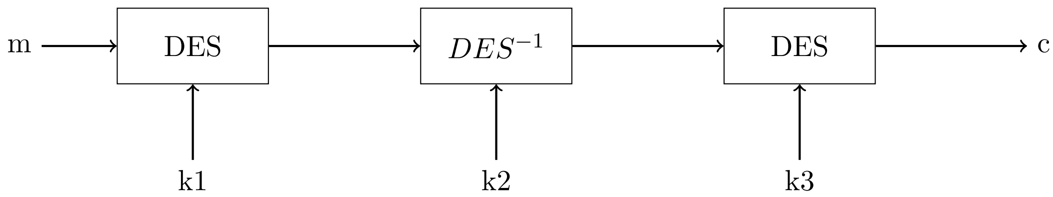
- byte Substitution(by passing thorough s-blocks and stuffs)
- Row shift (for diffusion)
- Mix column (for diffusion)
- Key addition(xor with key)

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:





RivestShamirAdleman (RSA) algorithm

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

- get two large prime numbers 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 - $a^{\phi(n)} \pmod n = 1$.

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 DiffieHellman 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

DiffieHellman 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(DiffieHellman 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, $a^{\phi(n)} \pmod n = 1$ 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.

Why are a and b chosen from {2,3,..,$p-2$}, i.e, 1 and $p-1$ are ignored?
If a = 1, $A = \alpha^1 \pmod p = \alpha$. Since A and α are known, its easily seen that they're equal and so $a = 1$ can be found. Then, $K_{AB} = B^a = B$ simply. Thus key get knowned.
If a = p-1, $A = \alpha^{p-1} \pmod p = 1$(Fermat's little theorem). Since $A = 1$, its easily seen that they're equal and so $a = p - 1$ can be found, provided α is a primitve root. Otherwise, a will be a factor of $p-1$ which also could be found easily ig.. Then, $K_{AB} = B^a$ simply. Thus key get knowned.

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)

Lecture - 10

Topic: Digital Signatures - RSA

Security Objective(CIA)

- Confidentiality is reqd. Encryption using a key ensures confidentiality in fact as they can be decrypted only people with the key.
- **Integrity:** Message shold not be tampered in between. We should be able to identify if its tampered or not.
- **Message Authentication:** It should be provable that only the intended sender sent the message and no one else.
- **Non-repudiation:** If someone sent a message, later it can be proved that none other than him/her sent it. Eg- signature. Like if signed, later they cannot deny that they didn't send it.
- **Access Control:** Who all can read the message, who all can modify it, etc should be able to be controled.

Digital Signatures

Similar to normal Signatures but are done on digital documents. It provides the above parameters including authentication and non-repudiation.

How to create digital signatures?

Say a document carrying message m is there and signature s. Message and signature are from "Message space" and "Signature space" resp (basically superset of messages and signatures). Person A is the owner and B checks the signature.
B calculates the sign of the document using key k i.e, $Sig(m)$ and checks if its same as S or not. If its same, then signature is verified and otherwise no.
If its document is tampered, calculated signature won't be same and could be identified that its tampered.
But this does not ensure Non-repudiation as anyone can find $Sig(m)$.

To include Non-repudiation, Assymetric Key distribution can be used, where private key is used to find S and it can only be verified by public key and signature cannot be calculated by it.

Digital Signature using RSA

Private info(i.e owner): d, p, q, $n = pq$, signature $S = m^d \pmod{n}$. Now, $e = d^{-1} \pmod{\phi(n)}$, n, and S are sent

Public info(i.e verifier): m ,n , S, e.
$S^e \pmod{n}$ is calculated and if its equal to m, then its verified and not verified otherwise.

What does attacker wants to do?:
Message is known to everyone(like everyone knows A send 1000 rs to C), but what an attacker wants to do is that, he wants to change the message to m' (say change 1000 to 10000 rs). Now, to calculate signature S', d is required. Say he made up some random d. Since, $\phi(n)$ is unknown, e cannot be calculated, and thus e can't be send.

Digital Signatures using Elgamal

- Setup Phase: p, α are choosed and are available publically
- Owner chooses d from $\{2,..,p-2\}$ and $K_{pub} = \beta = \alpha^d \pmod p$ is calculated and sent. This is called Long Term Public Key. This is not changed for every message the owner sends. Its maintained across all the messages he/she sends.
- Ephimeral key K_E chosen from $\{2,..,p-2\}$ such that $gcd(K_E, p-1) = 1$. Ephimeral keys are generated newly for each message the owner sends.
- Signature is calculated. like $r = \alpha^{K_E} \pmod p$ and $s = m - d.r \pmod p$. The pair (r, s) is the signature.

Known info - p, α, Public key β, message m, signature (r, s). **Verification:** is $\alpha^m \pmod p$ same as $\beta^r.\alpha^s \pmod p$? If yes, its verified.

Correctness

$$
\begin{aligned}
\beta^r.\alpha^s \pmod p &= \beta^r.\alpha^{m-d.r} \pmod p \\
&= \alpha^{d.r}.\alpha^{m-d.r} \pmod p \\
&= \alpha^m \pmod p
\end{aligned}
$$

Now, if someone wants to tamper the message, say they wants to create signature of the owner of different message m'.
$r = \alpha^{K_E}$ can be calculated by choosing some random K_E. But to calculate s, d is required which can't be solved from the eqation $\beta = \alpha^d \pmod p$ due to discrete logarithmic problem.

Disadvantage

The signature size is double the size of RSA as we are sending both r and s as signatures.

Lecture - 11

Topic: Digital Signatures Algorithm

Digital Signature Algorithm

This is currently widely used methof for Digit Signatures. This algorithm reduces the size of the signature from 2048 to 320 bits and 4096 to 422 bits as compared to Elgamal method.

- Choose p s.t. $2^{1023} < p < 2^{1024}$
- Choose q, a factor of $\phi(p) = p - 1$ s.t. $2^{159} < q < 2^{160}$
- Choose α s.t. $ord(Z_p) = q$.
  i.e, α is the generator of a subgroup of $\mathbb{Z}_p^*$ of order q.
  In other words, there exists a subgroup of $\mathbb{Z}_p^*$ such that $\{e : e^q = 1 \pmod p\}$ and α is the primitive root of the subgroup. We'll discuss this in detail later.
- $K_{pr} = d \in \{0, 1, 2, 3, ..., q - 1\}$.
- Choose $K_{pub} = \beta = \alpha^d \pmod q$
- Public parameters transfered to B are : $K_{pub} = \beta$, p, q ,α
- Ephimeral key $K_E \in \{0, 1, ..., q - 1\}$.
- Signature :
  $r = \alpha^{K_E} \pmod q$
  $s = (SHA(m) + dr)K_E^{-1} \pmod q$
  Assume SHA to be a funtion which converts any message to a 64 bit value. We'll discuss this later.
- SHA(m),(r,s) sent to B.

- Verification
- Compute

$$
\begin{aligned}
w &= S^{-1} \pmod q \\
\mu_1 &= w.SHA(m) \pmod q \\
\mu_2 &= w.r \pmod q \\
v &= (\alpha^{\mu_1}.\beta^{\mu_2} \pmod p) \pmod q
\end{aligned}
$$

- If $v \equiv r$, then the signature is valid. Else invalid

r, s are 160 bits each, and thus, the signature (r, s) is of 320 bits (much smaller as compared to Elgamal method of $1024 + 1024 = 2048$ bits.)

Correctness

$$
\begin{aligned}
s &= (SHA(m) + d.r)K_E^{-1} \pmod q \\
s.s^{-1} &= (SHA(m) + d.r)K_E^{-1}s^{-1} \pmod q \\
1 &= (SHA(m) + d.r)K_E^{-1}s^{-1} \pmod q \\
s^{-1}.(SHA(m) + d.r) &= K_E \pmod q
\end{aligned}
$$

So, $s^{-1}.(SHA(m) + d.r) = \theta q + K_E$.

$$\begin{aligned}
v &= \alpha^{s^{-1}SHA(m)} \beta^{s^{-1}r} \pmod p \pmod q \\
&= \alpha^{s^{-1}SHA(m)} \alpha^{s^{-1}r.d} \pmod p \pmod q \\
&= \alpha^{s^{-1}SHA(m)+s^{-1}r.d} \pmod p \pmod q \\
&= \alpha^{s^{-1}(SHA(m)+r.d)} \pmod p \pmod q \\
&= \alpha^{\theta q + K_E} \pmod p \pmod q \\
&= (\alpha^q)^\theta.\alpha^{K_E} \pmod p \pmod q \\
&= \alpha^{K_E} \pmod p \pmod q \ [\text{as } \alpha^q = 1 \pmod p] \\
&= \alpha^{K_E} \pmod q \\
&= r
\end{aligned}$$

Here too, as d is unknown signature can't be created by a random person other than the owner.

Clearing some concepts

Subgroup of $\mathbb{Z}_p^*$ such that $\{e : e^q = 1 \pmod p\}$ and α is the generator of the subgroup. Here, first notice that its a group on its own as the properties such as Closure, Associativity, etc exists.
Does inverse element exist?
We know that e^{-1} exits in $\mathbb{Z}_p^*$. We just need to prove e^{-1} is in the subset provided e is in it.

$$\begin{aligned}
(e^{-1})^q &= x \pmod p \\
e^q.(e^{-1})^q &= e^q.x \pmod p \\
(e.e^{-1})^q &= 1.x \pmod p \\
1 &= x \pmod p \\
x &= 1
\end{aligned}$$

Thus proved e^{-1} is in the subset. Thus proved that this subset is actually a group.
So just choose a primitive root α from this subgroup.

SHA is a hashing funtion which creates any message to a 64 bit value. It hashes so widely such that, practically every message hashes to a different value. Though through pigeion hole principle, there will exist messages to hashing to same value mathematically, practically we can assume they hash to different values.

Why do we hash the funtion? why can't we use the message at itself?
First, hashing increases confusion in message, i.e, even if just a small change is made in message, its hash would be completely different. Secondly, messages can be arbitarily long, and so to have it standard, we hash it to 64 bit value.

Lecture - 12

Topic: Digital Signatures Algorithm

Key Exchange

Two parties A and B wants to exchange key between them to use Symmetric Cryptography from then as Assymetric algorithm is computationally expensive to send the whole message.
Passive Eavesdropping: Just can listen on the common commutication channel between two parties and cannot send data.

Active Eavesdropping

Say an intruder is Active eavesdropper, i.e, he/she can listen to the common communication channel and also write or transfer messages.
Such an attack is called *Man In The Middle* attack. Take example of DHKE between people A and B. So, when $A = \alpha^a \pmod p$ and B are transfered from each other, the intruder reads them, modify them as A' and B' and send to each other.

A chooses a private key a from 2,3,..,$p-2$ Calculate public key $A = \alpha^a \pmod p$ Send A Recieve B'. $K_{AB'} = B'^a \pmod p = \alpha^{a.r_1} \pmod p$	R chooses a private key r_1 from 2,3,..,$p-2$ Calculate public key $A' = \alpha^{r_1} \pmod p$ R chooses a private key r_2 from 2,3,..,$p-2$ Calculate public key $B' = \alpha^{r_2} \pmod p$ Send A' to person B Send B' to person A Recieve A. Recieve B. $K_{AB'} = A^a \pmod p = \alpha^{a.r_1} \pmod p$ $K_{A'B} = A^a \pmod p = \alpha^{a.r_2} \pmod p$	A chooses a private key b from 2,3,..,$p-2$ Calculate public key $B = \alpha^b \pmod p$ Send B Recieve A'. $K_{A'B} = A^a \pmod p = \alpha^{a.r_2} \pmod p$

Both the parties can't even identify that such an intruder is present in between. Person A assumes B' was sent by actual person B and vice versa. Once, such a thing is done, the intruder can not only read the messages sent between them, but also can modify and send messages between them. Very similar attack is possible even in Elgamal. This problem arrises because, there is no authentication for any message. To resolve authentication, one may think we can use digital signatures. But, that is also prone to such a middle man attack.

Centralized Agency (CA)

Thus some authorised middle man is required to authenticate both and build trust. Such an agency is called Centralized Agency (CA). Such a system with CA is called PKI i.e *Public Key Infrastructure.* The above problem just requires people to send an ID along with message to authenticate.The centralized agency produces a certificate for whatever it sends and its ID. When we use https, we see a lock symbol in the browser. Its one kind of a certificate.
Now, for a person A to communicate, it asks for a certificate to CA by sending $(A = \alpha^a \pmod p), ID_A)$ (ID_A is some ID sent by it like IP address or serial number of processor or something which is unique across world). Now, CA produces a signature($S_A = Sign_{K_{private,CA}}(A, ID_A)$) on them using its own private key ($K_{private,CA}$) and sends it back along with its public key($K_{public,CA}$).

A chooses a private key a from 2,3,..,$p-2$ Calculate public key $A = \alpha^a \pmod p$ Send (A, ID_A, S_A) Recieve B. Check it with CA. $K_{AB} = B^a \pmod p) = \alpha^{a.b} \pmod p$

B knows the public key of CA and so it can verify (A, ID_A) using S_A. Thus B can be sure that its sent from A only and not from intruder. Now follow same steps for our regular algorithm.

Now, if intruder wants to do something, it can't give fake signature of CA about $A = \alpha^a \pmod p$ and ID as private key of CA is not known to anyone. Of cource if CA's private key is known, he/she can break anything on their way. Assuming its not possible, now communication is assumed secure.

Lecture - 13

Topic: Security Services

Security Services

- Confidentiality
- Authentication
- Integrity (Not tampered by third party)
- Non-repudiation
- Access Control
- ...

Confidentiality can be ensured by Encryption.
Authentication can be ensured by Digital Signatures.
Integrity can be ensured by Diffusion.
Non-repudiation can be prevented by Digital Signatures.

Access Control is our next topic of discussion :)

Elgamal

Vulnerable to Man in the Middle Attack.
Solution to this? Certification Authority as we saw in last class.
The signature of CA is obtained by providing Public key A and ID_A. Third party cannot obtain the certification in the name of A as their identity is verified at multiple levels like Govt ID, emails, phonecalls, etc. Also, the certification comes with a validity period, and should be renewed after that period. Nowadays, the certificate comes along with the device when it is bought.
There is no one centralised CA. The main CA is divided into multiple sub CAs. This is done as if the centralised CA is compromised, then all the data in the world is at risk.
This kind of security is used in SSL, TSL Protocols.
We have four levels of security:

- Application
- Transportation Layer
- Network Layer
- DLL
- Physical Layer

Lecture - 14
Topic: Security Protocols(Guest Lecture)

Have you ever noticed broken lock symbol in unsafe websites? What are these? How do we actually send messages?

Digital certificate

Every device would have a digital certificate which contains information like the person's identity(eg- name, address, organisation), public key, some serial number, validaity dates, certification authority's identity, its digital signature, etc which would be verified by the CA(Certificatation Authority). So indirectly, every user trusts the CA (or whoever is the issuer of the certificate). All the personal information would be stored in a data stucture, and certificate would be issued by it. But CA cannot certify all over the world right? Traffic would grow exponentially. So, CA can also allot people, and give a certificate that, this firm can issue certification. So end users can believe certificate issued by either CA, or by firms approved by CA. Actually, we can choose organisations that we trust for certification issuing. Many organisations, banks do this. Try this at room using the slides.

Protocol against Middle Man Attack

Say person Alpha wants to send message to Beta and there is a middle man Gamma trying to decrypt it. Each have a certificate from CA for their public key. With the public key of the CA, anyone can verify the certificate. Now, everyone knows everyone's public key, their cerificates. So if Alpha wants to send to Beta, he can use Beta's public key so that only Beta can decrypt it. When we make connection with google with https, we authenticate google using its certificate. See the slides, for SSL protocol where exchange of messages to google is clearly mentioned.
This is a broadcast system where each person can see anyone's message transaction with everyone having certificate and public keys. But, can an internet service provider(ISP like Jio/ Airtel) interfere in between? Actually, every internet packet is through ISP only, and so they can do anything they want. SSL certificates are for websites mainly. For emails, its seperate, for whatsapp, or facebook, it different it seems.

Lecture - 15

Topic: Security Protocols(Guest Lecture)

Revoking a certificate

Say you have your physical id card. And the institute wants to revoke it before the day of expiry. Since its physical, it can be broken, and revoked, but how to invalidate a digital certificate. We can't delete the certificate simply as its known by everyone. Also, if a person is offline, he won't Recieve the info that its revoked. We can't maintain a revocation list as there might be many certificates revoked everywhere, and its very difficult. There is a heirarchy of certification agencies which makes it more difficult.

Replay Attack

Say a car is unlocked by a key with IR transmitter. It can be listened, and replayed, or re created to unlock the car. This kind of attack is called replay attack. To avoid this, we can include time stamp along with the message.

So, see both the problems mentioned above. A certificate comes along with a freshness term to represent a time stamp at which a certificate is created.

Authorization / Access Control

We have seen before what both means. We need to somehow maintain permissions for each person to view/edit any data. There are two types of Access control:

DAC: Discresionary access control.
The owner has the full control over who can read the file, who can update/modify, etc.

MAC: Mandatory access control.
Irrespective of who is the owner, the sytem administrator has to enforce in system level. He need to design a policy and needs a mechanism to enforce it. There would be system, and people with an account in it can login on it. Based on his username or something, the system administrator should enforce that for that user, some of the files are available.

Subject(principle),Object(resources),Action

Matrix: Here we have a matrix with columns as objects, and rows as subjects(users). The matrix simply indicates the permission each user has over the files(objects).
Suppose user 2 wants to read file 3, the matrix says only write, so he cant read file 3. This Matrix is simply to maintain access control.
This is not an efficient way of maintaining access control - **Viru**

A more systematic way to maintain these permissions is by creating an access control list, for each file s list of the permissions each user have on it. The list can also be maintained capability wise, that is a list of permissions of particular user one every file.

Confused deputy problem

A compiler has read for

solution

Capability-wise list is safer because ... hmm... not sure

Lecture - 16
Topic: DAC and MAC

ACM(access control matrix)

This we have seen last class that permissions of every file and every user are stored in a matrix.

ACL(Access control list)

This is a big list which

Confidentiality bases access control model

Some of such models are:

Bella La Pudela model(BLP)

So we have subjects(users) S, and objects(files) O. Now, we have different levels of secracy of an object. They are - top secret, secret, confidential, and unclassified(public). Eg- some persons email is confidential, asc report is secret, and some other very important thing is top secret. These 4 categoried can be given to the objects. Then we assign a clearence label of a particular subject i.e, what all level files can he/she wants. Now, if a person has clearence level of top secret, then he littrally can see any resources they want. Coming down the heirarchy, the resources will be limited. This kind of policy is called "NO READS UP", i.e a person can't read files of higher sensitive information than their clearence level. Here, for writing, we follow something called "NO WRITES UP" where you can't write files of lower priveldge level. Thus, here, even though a public writes on a secret file, it won't be able to be read by other public people. Thus confidentiality is maintained. This may seem stupid as a public person has access to change secret info. But here, we don't care about other things, and we only care about confidentiality.

Integrity

In the previous method, we just implement "NO WRITES UP" and "NO READS DOWN". This can be observed to ensure integrity, i.e, un-authorised people can't write on secret files.

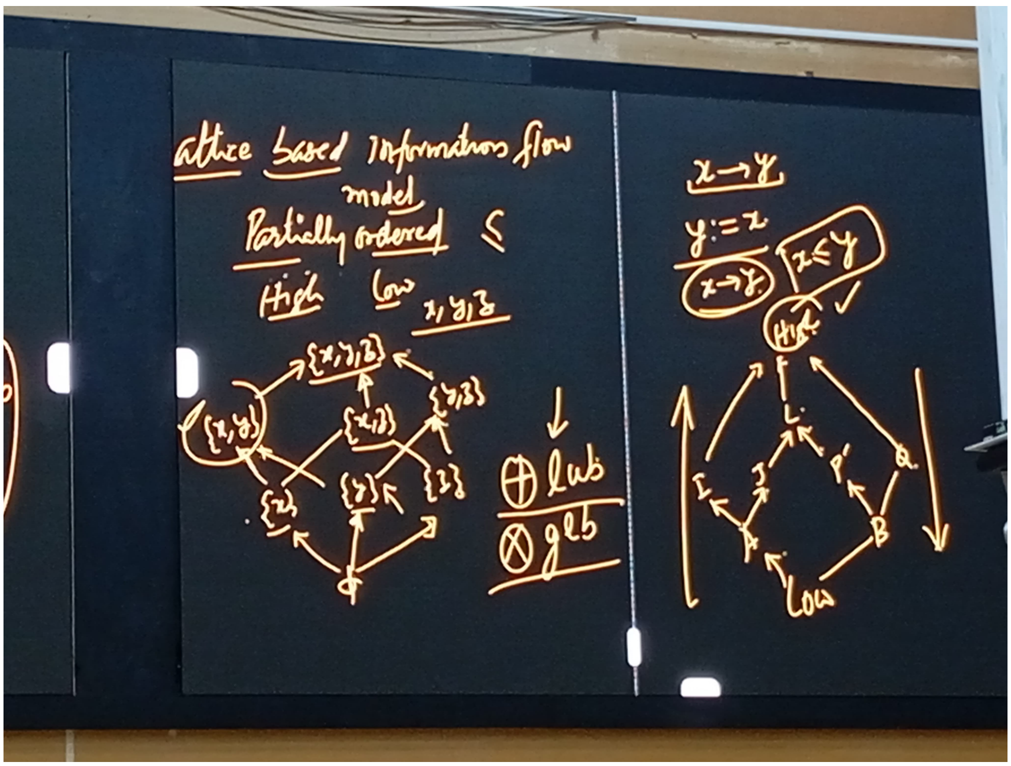
Lecture - 18

Topic: DAC and MAC

Previous class we discussed about information flow. There, we imagine a machine giving output according to input, and knowing the output, will we be able to find the inputs or not was the question.

Lattice based information flow model

Here, there is a direction of information flow, and based on whether its legitimate or not, etc, we can model it as a latice. We construct a partial order where the directed edges indicates the legitimate flow of information. Eg, if $x \leq y$, then its legitimate if information flow happens from x to y. We



Partial order

define two operators in latice, namely lowest upper bound ($\oplus$), and greatest lower bound ($\oplus$).

Example

Say we write $y = x + z, p = y + t$, so there is a info flow from $x \to y(x \leq y)$, $z \to y(z \leq y)$ but, there is no info flow from $y \to p$ [1].
Some of the properties of this model are :-

1. reflexive ($A \leq A$)
2. transitive. If info flows from A to B, and B to C, we can say it flows from A to C.

[1] This is because, there would have been a value of y which we know, but after its over written by x and z. So now using p, we can infer only about the new value, and not the previous one.

3. Anti-Symmetric. If info flows from A to B, and B to A, only possibility is that $A = B$.

RWFM

Its read write flow model.

Information Leaking

What if the algorithm is correct, but its implemented wrongly which leads to information leakage. This is called a Bug. We can identify this by methods like Static Analysis. We can dry run(i.e, mentally run it and see for implementation bugs) the code.

Lecture - –

Topic: Mid sem solutions

1. (a) integrity
   (b) availability, confidentiality
   (c) integrity / confidentiality
   (d) availability, confidentiality
2. 21 bits

Lecture - 20

Topic: Browser Security

Mostly, whatever website you login to, its asks for either its own login, or authentication, or google's account, etc. So, we believe google, facebook to keep our login passwords safely, and so we get in through google credentials. So now, what is the conversation, or policy between google and that particular website?
Google gives an API, and whichever website uses it, google lends its authentication to it. Also here, google is a witness for your activites in the website. It actually, now can see how many times you logged in, etc and so it can give suggestions accordingly.

Cookies

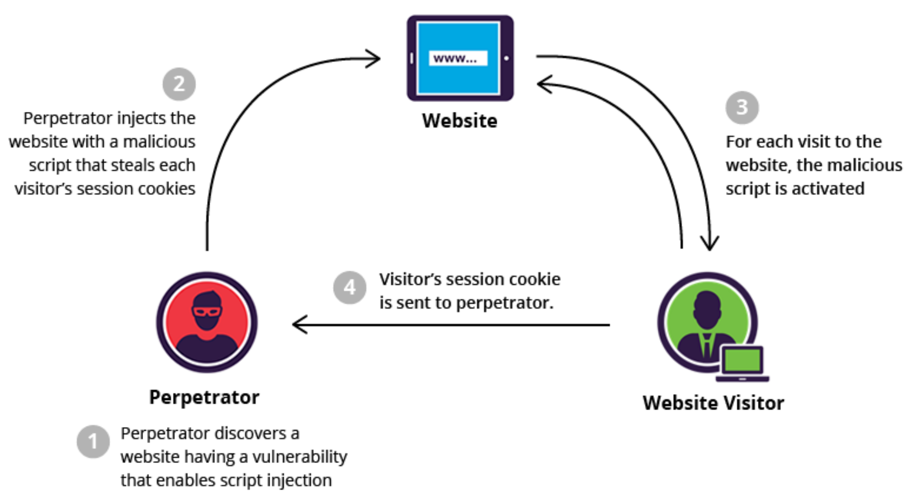
Its a mechanism to not ask the user again and again for our preferences. Eg- if you choose your preferred language, it'll be stored in cookie, and the next time you login, it knows the preference. Cookie, is a file, stored in a browser side to store such things. It may also contain your passwords. So is it possible for it to get stolen?

- If we use a shared computer, since the cookie is stored in the computer only, if another user uses it, it can get leaked.

Cookies are encrypted today to avoid the stealing.
Though they are encrypted, there are some ways to break it. They are

XSS(cross site scripting)

its a type of computer vulnerability typically executed with the help of web applications through breaches in user's web browser. Eg- if some website takes input, you can input a javascript code which modifies the javascript of the website and thus crashing it.



To avoid this, if we see some script like input, we ignore it. Nowadays, most websited so this, but an unaware web developer could be a harm for its user.

Reflected XSS

There are two methods of providing the input to the website. They are GET and POST methods.
In GET method, the information is directly given in the url and the web's code reads from the url and takes it as input and run accordingly. Eg- instead of searching for the string “data security” in google's tab, you can direcly give the url you're seaching for itself as https://www.google.com/search?q=data+security.
So, in this method, lets say a hacker gives you a link with input as a javascript, then if you click on it, the website starts working accoding to the hacker's contol.

CSRF

Typical practical scenario will be like, if you login in some website though your username, pwd, etc and you log out, the session should expire and authentication should end. But if csrf is not handled properly and you click back button, from your history, you can see again your personal data in website even after logging out.