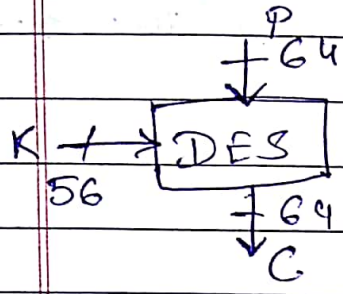
11.06.18 **CRYPTANALYSIS** (Prof. S. Mukherjee, IIT Kgp)



$E: \{0,1\}^n \times \{0,1\}^K \longrightarrow \{0,1\}^n$

One is to design the code (cipher text), the other is to attack the code.

Attack models:-

1. Ciphertext only attack - ciphertext is known to Oscar and he has to find the key.

$y = E_{DES}(m)$ $y \longrightarrow$

Alice ~~~~~ Bob

K = 56

↓

Oscar (Attacker)

2. Known plaintext attack -

Adversary knows some plaintext and the corresponding cipher.

Oscar knows $(P_1, C_1), (P_2, C_2), \ldots, (P_K, C_K)$

plaintext, ciphertext.

Goal of adversary:-

(i) get K (key)

(ii) get p^* from c^*

Session key - Key that changes very frequently (in minutes, second or micro seconds).

3. Chosen plaintext attack -

Suppose we give the adversary the Encrypt.exe (encryption machinary) a temporary access and the adversary can encrypt a chosen plaintext, get the ciphertext, guess the key & break the key.

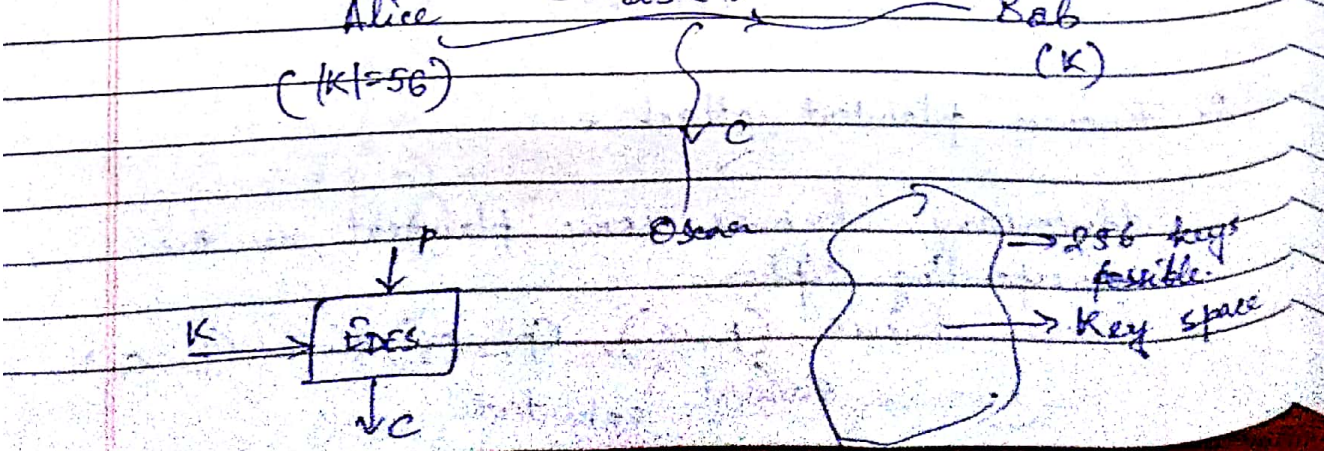
4. Chosen ciphertext attack -

We give the adversary temporary access to decryption machinary and the adversary can decrypt a ciphertext, get the plaintext, guess the key & break the code.

GENERIC ATTACK

1. Exhaustive Search

1. ~~Ke~~ Known plaintext attack - (P, C)



EFF - They made a hardware for parallel computing and broke DES in less than 3 days.

EFF - Electronics Founder Foundation

Worst case time = 2^{56} sec

Triple DES - They increased the key size (not that effect the performance as it requires 16×3=48 rounds).

Exhaustive search method - We try each and every possible key to break the code.

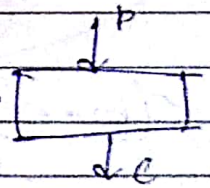
2. Table look up

We can mount this for known plaintext attack (ie we have to know both plaintext & ciphertext). → (p, c) → known.

$$E : \{0,1\}^n \times \{0,1\}^K \longrightarrow \{0,1\}^n$$

($\{0,1\}^n$: Plaintext Space; $\{0,1\}^K$: Key space; $\{0,1\}^n$: Ciphertext space.)

$$E_p : \{0,1\}^{K56} \longrightarrow \{0,1\}^{64}$$

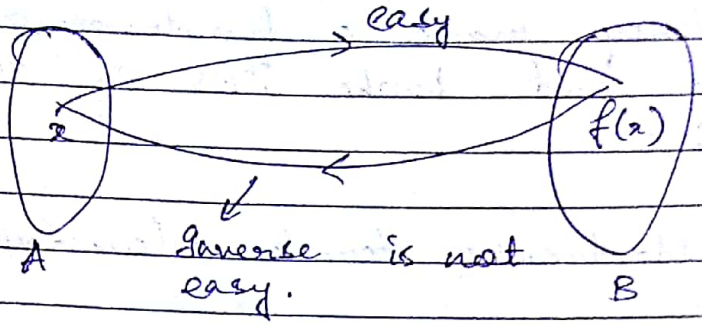


$$R : \{0,1\}^{64} \longrightarrow \{0,1\}^{56}$$

$$f : R \circ E_p$$

$$f(k) = R(E_p(K)) \; ; \; f : \{0,1\}^{56} \longrightarrow \{0,1\}^{56}$$

This a one way function (inverse is not easy). Encryption function should be polynomial time (easy).



Given Fig: One way function

Given $y = f(K)$, need to get K.

AES has 3 versions ~~depths~~ depending on no. of rounds.

→ 10 rounds
→ 12 rounds
→ 14 rounds.

We take x_1, compute $f(x_1)$ & store it.
→ ~~pre computation phase~~

$x_1, f(x_1) = y_1$
$x_2, f(x_2) = y_2$
$K \leftarrow x_K \quad = y_K$
$x_N, f(x_N) = y_N$

$N = 256$
$y = f(K)$

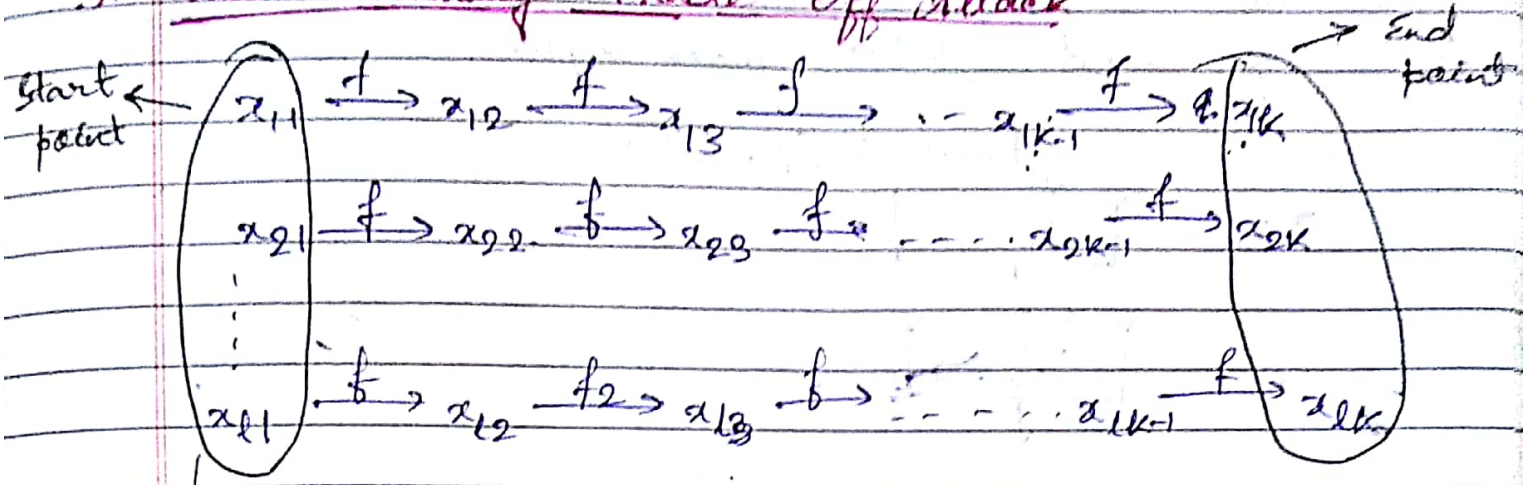
$O(\log N)$ → time complexity.

$N = 256$
$O(\log_2 2^{256}) = O(56)$

Offline time we don't consider into actual attack time.

Time-Memory Trade-Off Attack

3. Time - Memory Trade - Off Attack



→ This is called the Hellman table

$y^* \xrightarrow{f} y_1^* \longrightarrow y_2^* \longrightarrow$

Since we don't have much storage space so we only store the start point & end point.

Pre computation phase (Done in offline)

$(x_{11}, x_{1k}) \longrightarrow x_{j1}, x_{jk}$; $j = 1, 2, \ldots, l$

(x_{21}, x_{2k})

(x_{l1}, x_{lk})

There are many versions of this like rainbow attack etc.

Any block cipher or stream cipher can go under this attack.

This attack can mount for any cipher that can give us one way function.

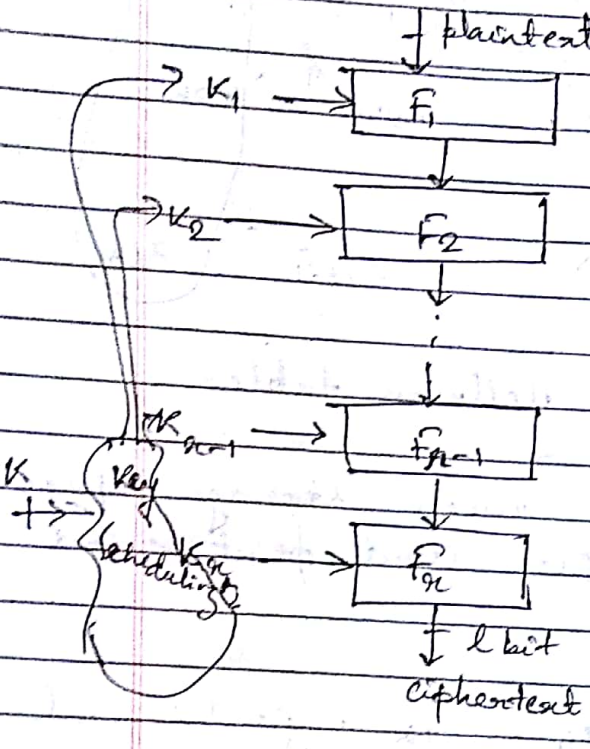
Date : / /

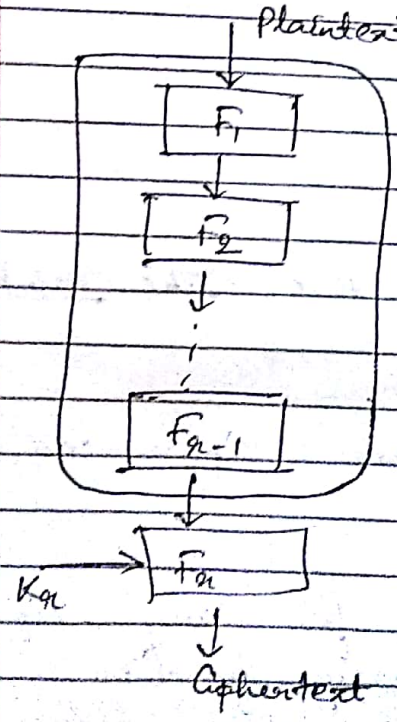
NON-GENERIC ATTACK

1. DIFFERENTIAL CRYPTANALYSIS

↳ Used for block cipher

n-round l-bit block cipher





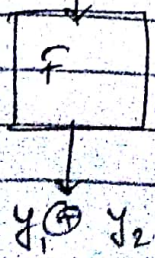
$f: \{0,1\}^l \to \{0,1\}^l$

differential with high probability

$\alpha \longrightarrow \beta$

If such α, β does not exist, we say that such differential attack does not exist.

$x_1 \oplus x_2 = \alpha$



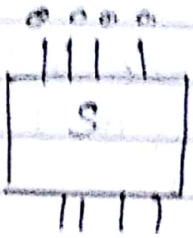
$y_1 \oplus y_2 = \beta$

α, β is predetermined.

If such α, β exist, then we can go for differential attack

S: $\{0,1\}^8 \rightarrow \{0,1\}^8$
is a function.

AES also uses S box (8 bit to 8 bit



goes from 16 bits

I/P → 0 1 2 . . . . 9 . . . A . . . F

O/P → . .

$k \backslash \alpha$	0	1	(2)	. . .	9	A	. . .	F
.								
.								
.								

↳ Compute this table as a C program.

Suppose $\alpha \rightarrow 2$

$x_1 \rightarrow$ 0 . . . 9 A . . . F

$x_2 = x_1 \oplus \alpha$

$x_2 \rightarrow$ . .

Compute y_1 and y_2

$k_1 = y_1 \oplus y_2$

Chosen Plaintext Attack

x_1 we are choosing arbitrary

and $x_2 = x_1 \oplus \alpha$

We have corresponding c_1 and c_2
We do not know y_1 and y_2

We choose a possible $K_x \rightarrow K_x^*$
(right K_x for which we get right b)
then we perform inverse.

$$f_x^{-1}(c_1, K_x^*) = y_1$$
$$f_x^{-1}(c_2, K_x^*) = y_2$$

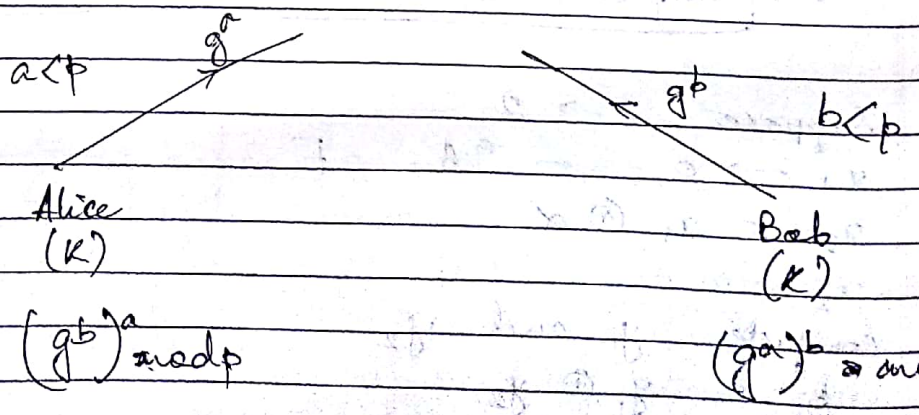
$$y_1 \oplus y_2 = b$$

We get this if we choose correct key.

PUBLIC KEY CRYPTOSYSTEM

Problems with Symmetric Key Cryptosystem

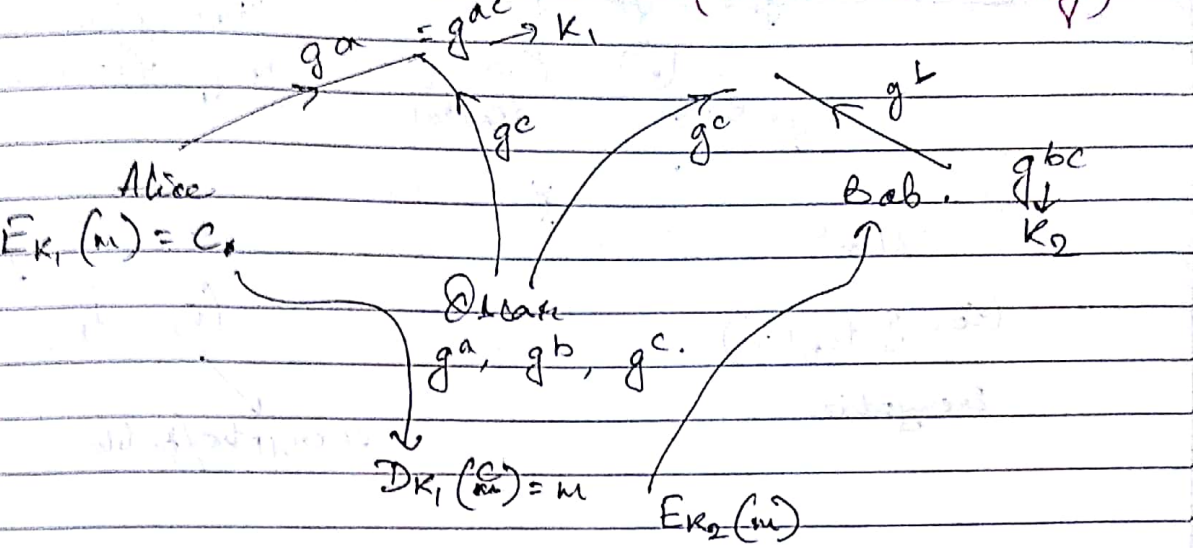
1. Diffie-Hellman



Given A

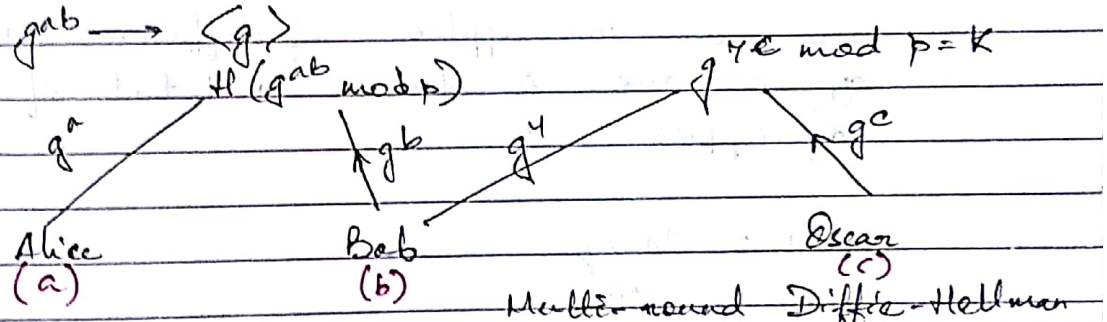
$g^a = A \rightarrow$ Getting a is HARD $\rightarrow$ Discrete log problem

Man in the middle attack (Active Adversary)



Oscar reads the message in the middle encrypts and sends to Bob. This is called man in the middle attack.

2. Sol: $g^{ab} \longrightarrow \langle g \rangle$



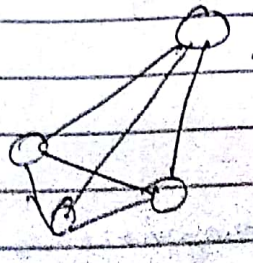
Multi-round Diffie-Hellman

Sol: $g^{ab} \longrightarrow \langle g \rangle$

$\langle g \rangle = (\mathbb{Z}_p, \cdot)$

Instead of multiround Diffie-Hellman for communication over a network, we can choose a different key for each channel.

∴ There are $^nC_2$ keys.



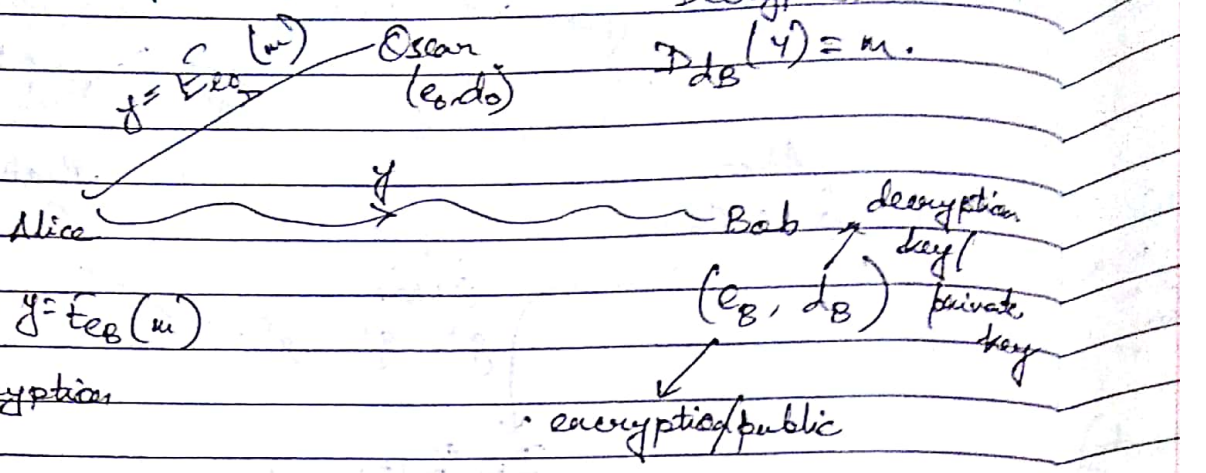
→ Complete graph

Nodes - users.

Edges - keys.

$^nC_2$ keys. = No. of edges.

1. Public Key Cryptosystem



2. Knapsack Cryptosystem

Knapsack problem is also called sub-set sum problem.

There are 100 apples of different weights, we have to choose a total of $S = 1000g$ of apples.

$a_1 = 200g$
$a_2 = 250g$
$\vdots$
$a_{100} = 150g$.

Input: $\langle S, a_1, \ldots, a_n \rangle$
$x_1, x_2, \ldots, x_n \rightarrow$ weights

$$a_1x_1 + a_2x_2 + \ldots + a_nx_n = S$$

$x_i = 0$ or 1

We use the Dynamic programming to solve this problem.
This is basically an exponential algorithm.

Suppose $\{a_i\}_{i=1}^{n}$, $i = 1 \ldots n$ is super increasing sequence.

$a_2 > a_1$

$a_3 > a_1 + a_2$

$a_4 > a_1 + a_2 + a_3$

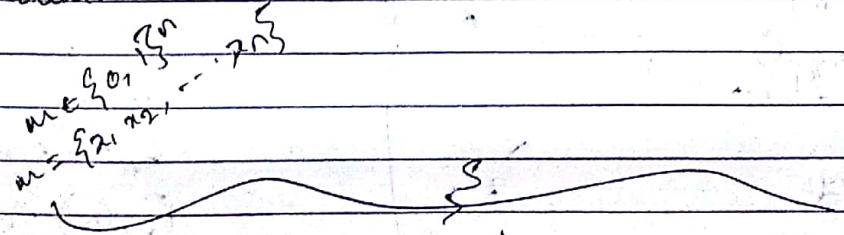
$a_i > a_1 + a_2 + \ldots + a_{i-1}$

$a_n > a_1 + \ldots + a_{n-1}$

Super increasing
Input: $\langle (a_1, \ldots, a_n), S \rangle$

1. For $i \leftarrow n$ down to 1
2. if $a_i > S$
3. then $x_i = 1$, $S \leftarrow S - a_i$
4. else $x_i = 0$
5. if $(S = 0)$
   return x_i
6. Sorry.

It is no more a hard problem, there is a polynomial time algorithm to solve this problem.



Alice
$e_B = (a_1, \ldots, a_n)$

$E_{e_B}(m) = \sum_{i=1}^{n} a_i x_i$

$= S$

Bob
$e_B = (a_1, \ldots, a_n)$
↓
Superincreasing
$\{S, (a_1, \ldots, a_n)\}$
$m = (x_1, x_2, \ldots, x_n)$

Oscar
$(a_1, \ldots, a_n)$
$x_i = 0/1$

This is easy knapsack cryptosystem.

$m = (x_1, \ldots, x_n)$ $\quad y = b_1 x_1 + \ldots + b_n x_n$

Alice

$e_B = (b_1, \ldots, b_n)$

$E_{e_B}(m) = \sum b_i x_i = y$

Bob

Set up

1. $a = (a_1, \ldots, a_n)$
2. $\gcd(w, m) = 1$, $m > a_n$
3. $b_i = a_i w \bmod m$

Setup phase - We can get the public key, private key pairs.

* 4. $e_B = (b_1, b_2, \ldots, b_n)$

$y = b_1 x_1 + \ldots + b_n x_n$

We take w^{-1} on both sides

$y w^{-1} = b_1 w^{-1} x_1 + b_2 w^{-1} a_2 + \ldots + b_n w^{-1} x_n$

$S = a_1 a_1 + a_2 a_2 + \ldots + a_n x_n$

$d_B = (\bar{a}, w, m)$

3. RSA Cryptosystem

Bob wants to send a message to Alice.

$e_A = (e, n)$

$d_A = (p, q, d)$

Alice

y

Bob

$P = \mathbb{Z}_n^*$

$e_A = (e, n)$

$E_{e_A}(m) = e^m \bmod n$

Ciphertext space $[\gcd(m, n) = 1]$

$C = \mathbb{Z}_n$

key space $K = \{(n, p, q, e, d) \mid ed \equiv 1 \bmod \phi(n)\}$

Setup

1. p, q two primes
2. $n = pq$
3. $\phi(n) = (p-1)(q-1)$
4. e, $\gcd(e, \phi(n)) = 1$
5. $ed \equiv 1 \bmod (\phi(n)) \Rightarrow d = e^{-1} \bmod \phi(n)$

$\phi(n) \rightarrow$ No. of integers less than n & co-prime to n.

$Z_n^* = \{ m < n \mid \gcd(m, n) = 1 \}$ ✓

$\phi(n) = |Z_n^*|$

$p = Z_n^*$
$\gcd(m, n) = 1$

Cryptosystem is a five tuple $\{p, c, k, e, d\}$.

Decryption: $m = y^d \bmod n = (m^e)^d \bmod n = m^{ed} \bmod n$

$y = E_{K_A}(m) = m^e \bmod n$
$= (m^e)^d \bmod n$
$= m^{k\phi(n)+1} \bmod n$
$= m^{\phi(n)k} \cdot m \bmod n$
↳ message

$ed = k\phi(n) + 1$ then only $ed \equiv 1 \bmod \phi(n)$

Euler's Theorem:

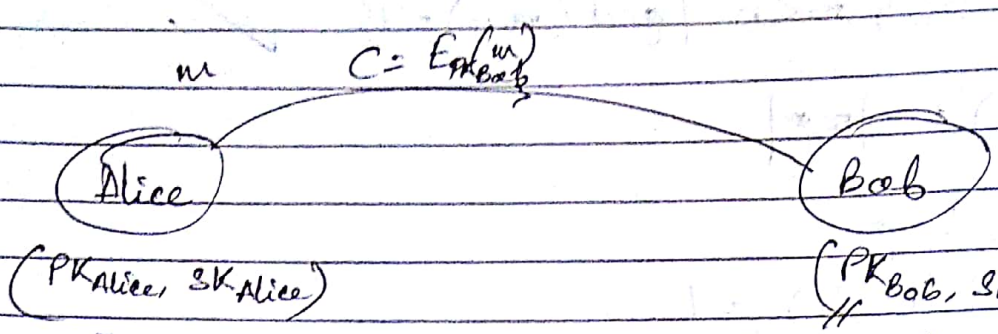
$$m^{\phi(n)} \equiv 1 \bmod n, \quad \gcd(m, n) = 1$$

2nd Half

IDENTITY-BASED CRYPTOGRAPHY

– Prof. Ratna Dutta, IIT Kgp (Dept. of Maths)

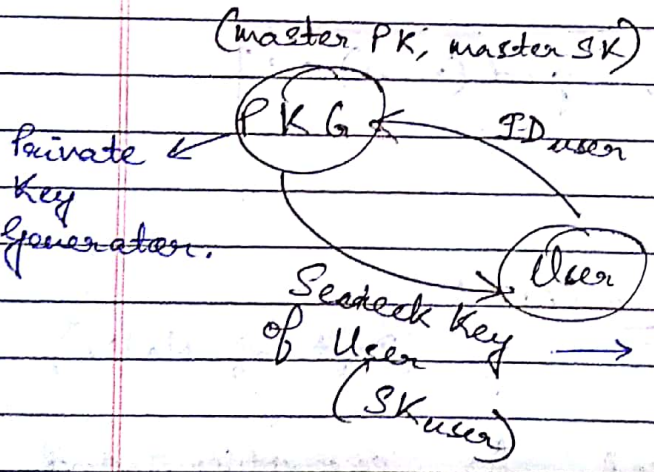
- Public keys are used for encryption and digital signature verification.
- Verification of digital signature is done publicly.
- Private keys are user specific, should be kept secret.
- Before using a public key, one should verify the claimed authority.



Alice encrypts the message using Bob's public key, only Bob can decrypt the ciphertext ~~because~~ using his secret key as Bob's public key has been used to ~~en~~ encrypt.

Identity-Based Public Key

Public of Bob (PK_{Bob}) = Identity of Bob $ID_{Bob} \in \{0,1\}^*$ ↓ Some random string.



Secret key → Secret key is issued on the flight.

Cryptographic Bilinear Map

$G_1 = \langle P \rangle$
DLP in additive group
given (P, aP) find a.

$G = \langle g \rangle$ multiplicative
given g, g^a find a.

$e(P_1+P_2, P_3) = e(P_1, P_3)\, e(P_2, P_3)$

↳ Bilinearity
Non-degeneracy.

Date : / /

Three-Party Key Agreement - see slide.

A → a
↳ Secret key of A.

$Key = e(bP, cP)^a = e(P, P)^{abc}$

$\hat{e} = e$, $\hat{e}$ and e are same.

Publicly available → $\langle P, aP, bP, cP \rangle$

↓ Compute

$e(P, P)^{abc}$

This is called the Bilinear Diffie-Hellman problem.

SOK Key Agreement

Setup:

(G_1, G_2, e)

$e : G_1^2 \rightarrow G_2$

$G_1 = \langle P \rangle$; $|G| = q$

(A) $SK_{IDA} = sH(ID_A)$

(B) $SK_B = sH(ID_B)$

$mSK = s$, $mPK = sP = P_{public}$

PKG → ID_u → user

SK_u

$H : \{0,1\}^* \rightarrow G_1$ (H → Hash function)

$H(ID_u)$; $SK_{ID_u} = sH(ID_u) \in G_1$

Email id is a public key, we are using hash function to map it as a point on G_1.

A computes $= e(SK_{ID_A}, H(ID_B)) = e(H(ID_A), H(ID_B))^s$ ($SK_{ID_A} = sH(ID_A)$)

B computes $\rightarrow e(SK_{ID_B}, H(ID_A)) = e(H(ID_B), H(ID_A))^s$ ($SK_{ID_B} = sH(ID_B)$)

$$e(P, Q) = e(Q, P) \quad \forall \; P, Q \in G_1$$

↳ This is the property of the bilinear map.

- Hash function is public
- ID is public
- So $H(ID_A)$ & $H(ID_B)$ can be easily calculated.
- But to find 's' (master secret key) is a hard Discrete log problem over G_2

Identity-Based Encryption (IBE) – see slide.

Motivation: To simplify certification management. No need to store public keys as public keys are identity of the user.

Boneh-Franklin (2001) IBE Scheme (see slide)

Encryption: $E(pk, m)$
Decryption: $D(sk, c)$.

Alice Bob

$e: G_1 \times G_1 \rightarrow G_2$ $\langle P \rangle$ is the generator of G_1
$\langle P \rangle$

$Q_{ID} = H_1(ID) \in G_1$

$m \oplus H_2(e(Q_{ID}, P_{pub})^x)$

Hash function
$H_2 : G_2 \rightarrow \{0,1\}^n$

$H_1 : \{0,1\}^n \rightarrow G_1$

$E(pk, m) = (\underbrace{xP}_{u}, \underbrace{m \oplus H_2(e(Q_{ID}, P_{pub})^x)}_{v})$

$x \in_R Z_q^*$

$D(SK_{ID}, c)$, $SK_{ID} = sH_1(ID)$

$C = (u, v)$
↳ Ciphertext is a function of u & v.

$e(Q_{ID}, P_{pub})^x = e(\underbrace{sQ_{ID}}_{= SK_{ID}}, xP)$

$P_{pub} = sP$
$Q_{ID} = tP$

The steps:

1. Setup
2. Extract
3. Encrypt
4. Decrypt.

Decryption:
$V \oplus H_2(e(SK_{ID}, U)) = m.$

$\Rightarrow m \oplus H_2(e(Q_{ID}, P_{pub})^x) \oplus H_2(e(SK_{ID}, u))$
$= m.$

$e(Q_{ID}, P_{pub})^x = e(P, P)^{tsx}.$

$\langle P, tP, sP, xP \rangle \rightarrow e(P, P)^{tsx}.$
This scheme is based on bilinear Diffie-Hellman problem.

Random Oracle - We assume that Hash function (H_2) is truly random → This is theoritically possible but not practically.

Boneh-Boyen's IBE Without Random Oracle

See slide.

G_1, G_2, e

$ID \in \mathbb{Z}_q^*$

message space = G_2

PKG: $mPK = (U = xP, V = yP)$, $mSK = (x, y)$

ID → User

$S_{ID} = \left(x, \frac{1}{ID + x + xy} P\right)$ → We choose this function because they found it to be secure according to the given problem statement

M → Denotes message in slide

$Enc(ID, m) = C = \langle \underbrace{s(ID)P + sU}_{X}, \underbrace{sV}_{Y}, \underbrace{e(P,P)^s \cdot m}_{Z} \rangle$

Choose s randomly from $\mathbb{Z}_q^*$

Ciphertext is a three tuple

$mPK = e, V.$

$Dec(SK_{ID}, C) \rightarrow \frac{Z}{e(X + rY + K)}$

$\left(x, \frac{1}{ID + x + xy} P\right)$ (X, Y, Z)

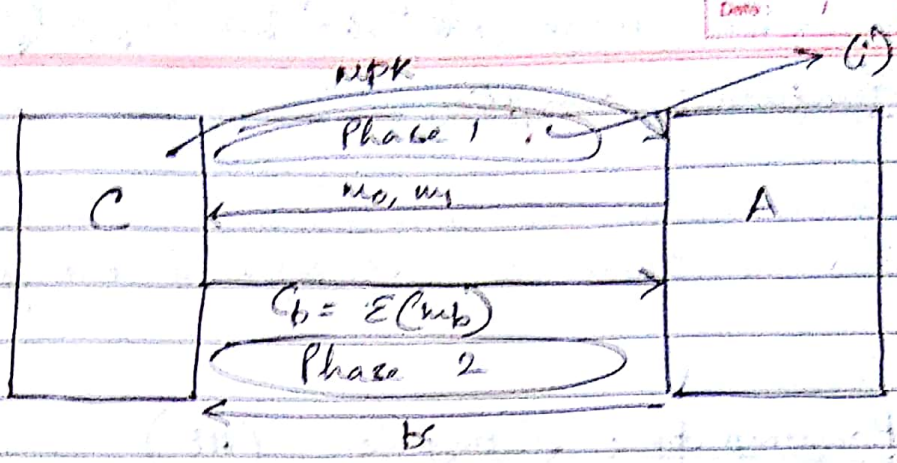
$e\left(s(ID)P + rsV, \frac{1}{ID + x + xy} P\right)$

$e\left(IDP + xP + xyP, \frac{1}{ID + x + xy} P\right)$

Security Model for PKE

IND - ID CPA on Semantic Security.

Chosen Plaintext Attack.



(i) $ID_i \neq ID$

B → A → For chosen plaintext

I_{ID_i} → For getting the secret key.

(ii) Decryption queries

$(ID_i, C_i) \neq (ID, C_b)$

B → A → Target ciphertext.

m_i

Adversary is forbidden to query that target ciphertext.

We are making our adversary more powerful to ensure the security.

Selective Security Model

The only difference is initial phase is before phase 1.

Fuzzy IBE

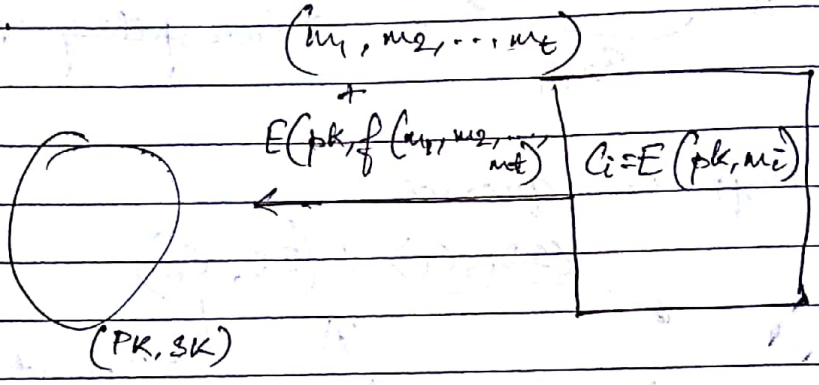
If out of 100 characteristics, 10 match, then the system will allow us to decrypt.

Functional Encryption

Generalisation of public key encryption.
This a quite a hot topic in encryption.

Homomorphic Encryption (HE)

Performs homomorphic evaluation on the given ciphertext.



f is some function → + or.

There are several encryption schemes that are partially homomorphic.

Fully homomorphic encryption scheme was first realised using lattices.

Evaluation - Allowing some untrusted server to perform some expensive computational operations. Like say we want to know the standard deviation of the the data, we don't have much computational power to do it so we ask some server to compute the standard deviation on the encrypted data and give us the result in encrypted form so we can easily decrypt it.
Here standard deviation is just an example of f.

Evaluation is very important in CLOUD COMPUTING.

Functional encryption is used extensively on cloud computing and messaging apps like WhatsApp.

KeyDer → just like giving ID & getting SK.

Functional encryption is an active research area in public key cryptography.

Inner product functional encryption.

Slide 34 – Alice (0.6, 0.3, 0.1) → weight vector.

El Gamal Encryption Scheme.

Email – ratna.dutta@gmail.com.
Book – Dr. Abhijit Das – Public Key Encryption