02/08/2024

Information Securities

confidentiality (C)

Integrity (I)

authenticity (A)

(A) —m→ (B)

sender — receiver

only cryptography is not sufficient

→ access control
→ authorization
→ availability

Cryptography Algo

- no mechanism to prevent attackers to modify the message
- receiver should be able to identify the message if changed

(I)

Algo Vs protocol

min 2 computer req

min 1 is enough to run algo

you should be able to recognise the actual sender has sent the message

(A)

- In some spcl case ($K_1 = K_2$) — inverse of the key K_1 is K_1 itself
- In general there will be two diff keys

Non Repudiability (4)

Sender should not deny that they have sent the message

Cryptographic Algo

symmetric ($K_1 = K_2$)

asymmetric ($K_1 != K_2$)

(2 diff algo for sender & receiver)

Only Cryptography →

- Confidentiality
- Integrity
- authenticity
- Non Repudiability

Four classes of cryptography

Encryption / Decryption (C) (Confidentiality) → can be both asymmetric / symm

Hashing

MAC → only one algo is used by both S & R

(always symmetric)

Digital signature

(2 diff algo for S & R)

attackers can always modify the message

(DOSS → availability requirement)

buffer overflow

such that real request is also ignored

Authorization protocol access control → purpose

For Hash => No Key required (Keyless)

always asymme

plain text data → cipher text

(encryption)

Why key is needed while encryption
→ to ensure that only receivers key will be able to decrypt but not others key

cipher text is then sent to qonchannel

Ciphertext → plaintext

Decryption Algo

(string of bits)
size also might be same

(OR)

$D/E : \{0,1\}^n \times \{0,1\}^l \rightarrow \{0,1\}^m$

n bits → space for plain text

l bits binary string → Key

space for cipher text

(Encryption & Decryption Algo) → public

extra

secret piece of strings of information → only known to sender & receiver

Key

needs for encryption & decryption

only known to intended receivers

size may be > (or) < $E(m, K_1) = c$

$m \in \{0,1\}^n$

$K \in \{0,1\}^l$

$D(c, K_2) = m$

$K_1 \leftrightarrow K_2$ inverse

(some relationship)

K_1, K_2 are inverses of each other

Key is not same to password

(Human can remember) (we create)

(must be random) sequence of bytes

(Human can't remember) (generated by computer)

(Key ⇐ Hash of password)

5|08|2024

Group ∀ for $a \in S$

i) closure
ii) associativity ⇒ $a \cdot b = b \cdot a$
iii) identity ⟹ $a \cdot e = a$
iv) inverse ⟹ $a \cdot b = e$

$\langle Q, + \rangle$ ✓
$\langle Q, \times \rangle$ ✓
] → groups

$\langle Z, + \rangle$ ✓
$\langle Z, \times \rangle$ ✗
$\langle R, + \rangle$ ✓
$\langle R, \times \rangle$ ✓
} groups

Groups with commutative ⇒ abelian group

field ⇒ Group with multiplication
&
Group with addition $\langle Z, +, \times \rangle$ → not field

$\langle R, +, \times \rangle$ → field

Ring ⇒ same as field (but multiplicative inverse is not mandatory)

$\langle Z, +, \times \rangle$ ⟹ Ring

Infinite.

closure property ⇒ define domain

Finite group / field ⇒ for cryptography

$G = \langle \{0,1\}^n, \oplus \rangle$

binary strings ↑ (XOR) operation

identity ⇒ all zeros

inverse ⇒ same string

$K_1 == K_2$

purpose of Hash

(Assymethic cryptographic ⇒ public key cryptographic).

sender can be anyone

So K_1 should be announced to all

(Not kept secret to anyone)

shared secret

Private key only known to receiver

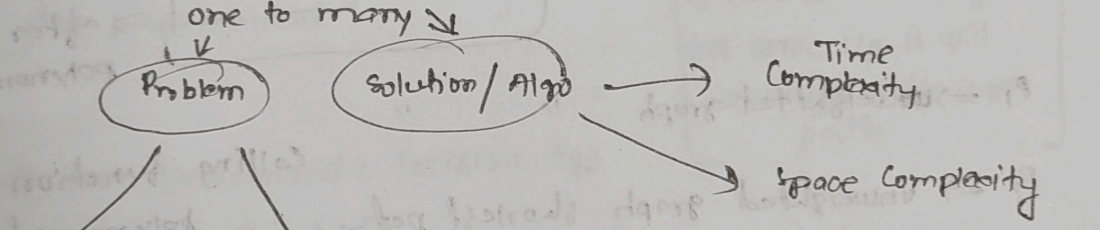
algorithms are all publicly known to all

(Public and private key are inverse to each other)

If public key is known to all and public key is inverse of private key then why can't we calculate the private key ?

✗ i) may be inverse does not exists. ✗ (always inverse exists)

ii) It's hard to calculate the inverse



Easy

Hard ⇒ (Can't be solved in polynomial time by known algo)

n ⇒ size of inputs (units: no. of bits req. to hold the i/p data)

(not the value of the i/p)

time complexity

but not the

complexity of the algo

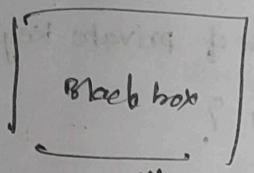
NP (diagram check karna hai)

unsolvable problems

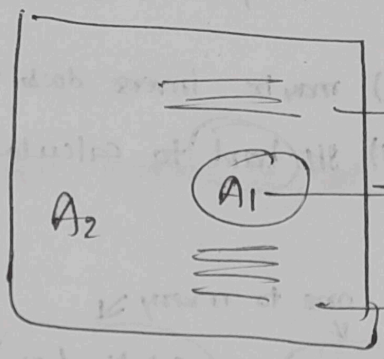
A harder than P_2 ⇒ P_2 reduces to P_1

Problem reduction ⇒ ?

using the soln of P_2, P_1 can be solved



give input get output
don't know how working
but can be used
any no. of times



P_1 ⇒ weighted graph " "

P_2 ⇒ unweighted graph shortest path

Calling function polynomial no. of times

using P_1 soln P_2 can be solved
is polynomial time
P_2 reduces to P_1

by giving c weights to all ⇒ unweighted ⇒ weighted

(before calling need some execution)

If A_1 is easy then A_2 must be easy
because we made constrains such that
(A_2 is hard as long as A_1 is hard)

9th Aug. 2024

(Non negligible success prob)

↓

(function of i/p size)

≠ asymptotic time complexity ?

Pseudo Random generator function

Hard : there need not be non deterministic poly algo

Problem

Polynomial soln (Easy) | No polynomial soln (Hard)

Problem

Deterministic ⇓ [same i/p then always same o/p] → Non deterministic → different

Smooth rate very high (expo)

Randomised Quick Sort ⇓ (Non Deterministic)

[uses random nos to choose the pivot element] → [source of Randomness / No randomness] → Yes

[If there is branching and for same i/p it will always take same path] → may be different

≠ Non Deterministic Decision Algo

L.V | M.C

P Vs NP Vs NP complete Vs NP Hard

↓ In Crypto

Yes biased

If it says yes, it is always correct

If it says NO, it may not be correct

No biased

If it says NO, it is always correct

there should be a candidate soln in polynomial time checking

(alternate)

NP ⇒ It does not have deterministic polynomial time algo but it must have non deterministic polynomial time algo

my need = attackers problem

In Cryptography security

It should be hard

encryption, decryption, hash, map ⇒ they should be easy problem

i) only correctness proof is not enough

for any message $M \xrightarrow[K_1]{} M'$ then $M' \xrightarrow[K_2]{} M$

(K_1 and K_2 inverses)

ii) soundness also needed ⇒ problem for the attacker is hard

⇓

≡ [there should not be any deterministic polynomial time algo]

≡ [and if there exists non deterministic poly then its prob of success is negligible]

one-way function ⇒ inverse cal. is hard

Trapdoor function ⇒ but if you know the secret then its inverse can also be calculated

Integer factorization problem

iterate constant no. of time

$36 = 1 \times 2^2 \times 3^2$

21, 33 → product of only two primes exactly

Computation (or) decision problem in cryptography Formal

given: n

find / compute p

s.t p/n

given n

decide: if n is prime

16/08/2024

Public Key of B

K_e = Inverse (K_d)

↑ Private Key of B

for symmetric Encryption

$K_e = K_d$

C Encryption Decryption

A sender

B Receiver

may be sym / asym

$\{0,1\}^l$ $\{0,1\}^n$ $\{0,1\}^l$

m → [E] —C (ciphertext)→ [D] → m

plain text

$\{0,1\}^n$ K_e K_d

n and l are almost same

may be 1gb

E, D ⇒ Encryption / Decryption

I

Hash → Keyless

neither symm nor asymm

A sender $\{0,1\}^n$ m → Hash → h → (||) → m||h

B Receiver m → Hash → h' ; h == h' ? 1 accept : 0 Reject

n >>> l

128 bits

A-I

MAC ⇒ symmetric

m → MAC (K) → d → (||) → m||d

K → MAC → d'

d == d' ? 1 accept : 0 Reject

Receiver

sender

A-I-N

assymetric

sender A K_{sign} m → Sign → s → (||) → m||s

B Receiver K_{verify} m, s → verify → 1 / 0

Private key of A (K_{sign}) = Inverse (K_{verify}) → Public key of A

Integrity without authenticity is meaningless

MAC ⇒ A, I

Signature ⇒ A, I, N

Encryption, Decryption ⇒ Authenticity only

Hash ⇒ only I (SHA I, SHA (II))

Correctness not enough alone
Soundness relationship with security

attackers objective opposite of users → C, I, A, N

$C = E(m, K)$ In case of Symm Encryption

data confidentiality, key confidentiality

Soundness Req. of Hash

→ no key ⇒ confidentiality of Hash no need

what should be the objective of the attackers

It should be a computation problem not decision problem

first define I/p & O/p

m, h is given and hash algo also given
need to calculate h'

this problem must be hard

try to compute m' such that $H(m) = H(m') = h$

hash is many to one function

mathematically possible finding m' this problem should be hard

$2^n \longrightarrow 2^l$ where $2^n >> 2^l$

given m ⇒ h calculated
find another m' such that its hash also same as h

Collision Resistance — finding m' is computationally hard but it is not impossible

Diffusion property

MAC → Inside we uses hash algo

Random Oracleness

each digest value is equally likely ⟹ then problem is hard

Hashing ⇒ Deterministic algo
↓
not a Randomized algo

Infinite Groups

Finite Groups, field, Ring } no. of element in the set is finite

$Z_n \Rightarrow 0$ to $n-1$

$G = \langle Z_n, \pm \rangle$ finite additive group
↓
mod n

$3+5 = 8$
$3+7 = 0$
$3+8 = 1$

identity ⇒ 0
inverse ⇒ (n − a)

$b = n-a$

binary

System crypto faster than asym

↓

equally secure both of them

for multiplication

$(3 \times 7) \bmod 10 = 1 \Rightarrow (3^{-1}) = 7$
$(7^{-1}) = 3$

only for co-primes ⇒ inverse will exist

{1, 3, 7, 9} Z_n^* lm n but also co-prime in Z_n

$\langle Z_n^*, \times \rangle$ ⇒ valid multiplicative group

$F = \langle Z_p, +, \times \rangle$ finite field

Galois field

Galoi field

Finite field defined over prime nos for

Why do we need asy crypto when we have faster symm crypto

{ data ⇒ symm encryption
  its key ⇒ asym encryption }

- non repudability can not be achieved using symm Encryption
- Key distribution problem
  → If an attacker intercepts the key, they can decrypt all the communication
- assym. encryption ⇒ do not required secure channel to exchange

Hard problem Integer factorisation

Discrete log Problem (DLP)

$a^0 = 1$
$a^1 = 5$
$a^2 = 3$
$a^3 = 4$
$a^4 = 9$
$a^5 = 1$
$a^6 = 5$
$a^7 = 3$
⋮

$G = (\mathbb{Z}_p^*, \times)$

$p = 11$

$i = 5$

$\{1, 3, 4, 5, 9\}$

$i \rightarrow$ order

$a^i \bmod p = 1$

$i = 5$

order of $(a) = 5$

$2^0 \longrightarrow 1$

$2^1 \longrightarrow 2$ — $p = 11$

$2^2 \longrightarrow 4$

$2^3 \longrightarrow 8$

$2^4 \longrightarrow 5$

$2^5 \longrightarrow 10$

$2^6 \longrightarrow 9$

$2^7 \longrightarrow 7$

$2^8 \longrightarrow 3$ — 10

$2^9 \longrightarrow 6$

$2^{10} \longrightarrow 1$

will be generated using two so they are called generator

1, 2, 3, 4, 5, 6, 7, 8, 9, 10

generators / primitive roots

ith root of the identity element

1, 3, 4, 5, 9

$2 \rightarrow \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$

$5 \rightarrow \{1, 3, 4, 5, 9\}$ ⟹ cyclic subgroup

any element of this group can generate this group

generates a valid group under the same operation

subgroup — subring — subfield

✗ super group

group extension field / ring

$g^{\alpha} \bmod p = \beta \longrightarrow Z_p^*$

$\alpha \longrightarrow Z_p^*$

g : generator

$g_p^{\alpha} : Z_p^* \longrightarrow Z_p^*$

bijective function $\Rightarrow$ Inverse Exists

$g^{\alpha} = \beta \Rightarrow \alpha = \log_g \beta$

$g^{\alpha} \bmod p = \beta$

$\alpha = d\log_{g,p} \beta$

} Calculation easy or not

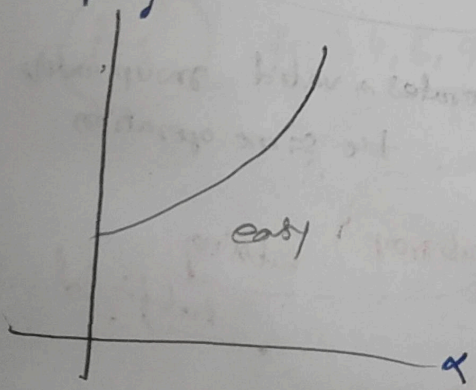
Diffie - Hellman Key exchange algorithm

→ used to exchange cryptographic keys over a public channel

security relies on

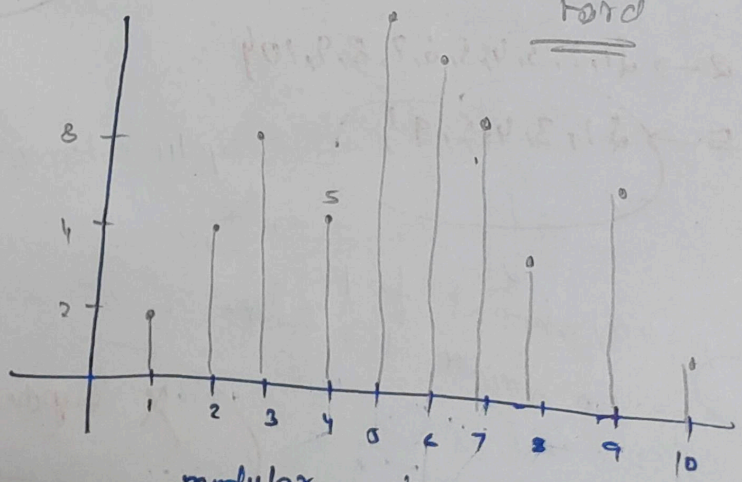
Why hard in Discrete case finding log

$\beta = g^{\alpha}$



normal

pattern



modular arithmetic

→ no pattern

↑ (or) ↓

uses OpenSSL library to generate a pair of RSA keys for a certificate authority

Alice

Bob

1) generate Puplic key & Private key

1) Key distribution and managment
2) scalability
3) Authentication and Trust
4) Non Repudiation

a man in the middle attacle

CA → acts as trusted third party

Port no.

80

443

HTTP → insecure

HTTPS → secure

uses SSL and TLS

30/08/2024

Diffie Hellman Key exchange Algo

given g, p, α } easy
find β

$\beta = g^{\alpha} \bmod p$

↓

something like binary search

D.L.P

f^{-1}

given β, g, p] hard
find α

$\alpha = d\log_{g,p} \beta$

→ No pattern here

→ Can't put any algo to find

Search domain remains same every time

→ exhaustive search

→ exponential complexity

Hardness assumption

↓ assumption

there exists no polynomial time algo till date

dLP, Integer factorisation ⇒ Hard problem

as long as Integer factorisation is hard, RSA is secure

$a \leftarrow Z_p^*$

$G_1 = \langle S_1, \cdot \rangle$ $G_2 = \langle S_2, \cdot \rangle$

1) same operator (multiplication modulo p)
2) set is the subset element of Z_p^*
3) find subset $S_2 \leq S_1$ like this

$\beta = g^{\alpha} \bmod p$

α → element in 1 to q (Z_q)

β → element in Z_p^*

$\{ a^1 \bmod p, a^2 \bmod p, a^3 \bmod p, \ldots, a^{q} \bmod p \}$ → S

$q = |G_1|$

order of G_1

difficulty depends on q not p

p must be large then only q can be large

order q would be a divisor of

(q must be divisor of (p-1))

↓ size of main group

↓ must be a prime no.

{we expect order of group to be large prime no}

↓ field operation can not be performed in exponent domain

Cryptopp

→ take random no

→ find order

1024 bits

→ if order is large then okay

else discard and use other random no.

prime subprime generators

p, q, g → crypto++ library

finding generator of any group is a Hard problem

subgroup generating → polynomial

160 bits

Diffie Hellman Protocol

→ two party key agreement protocol

→ using symmetric encryption (Data encryption)

→ Key → assymmetric

→ channel is semi trusted

two aspect of security

A, Integrity

authenticity.

but not confidentiality in the channel then no need assym. key needed

Authenticity, Integrity, Confidentiality → symmetric

✓ ✓ ✗

↓ assumption

semi trusted

→ otherwise, from where I am getting your public key they can change the public key itself first.

→ Man in the middle attacker

Session Hijacking

assuming No middle man attacker (1)

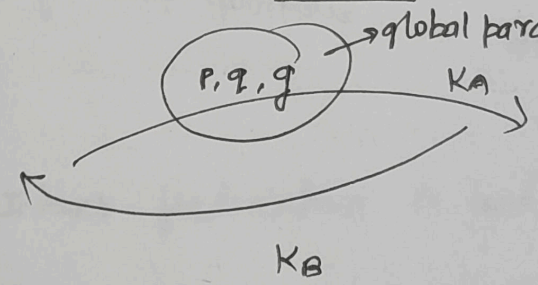
Public key certificate authority (2)

TCP → HTTP Vs HTTPS ← SSL / TSL

P.K.I
- certificate of web
- list of trusted certificate authority
- import/export How to a certificate in my browser

certificate Revocation

31/08/2024



i) $a \xleftarrow{R} Z_q^*$ $\quad b \xleftarrow{R} Z_q^*$

ii) $K_A \leftarrow q^a \bmod p$ $\quad K_B \leftarrow q^b \bmod p$

Diffie Hellman public key exchange

iii) a, b → private key of A, B

K_A, K_B → public key " "

$(K_B)^a \bmod p$ $\quad (K_A)^b \bmod p$

$(K_B)^a \bmod p = (g^b)^a \bmod p = g^{ab} \bmod p$

$(K_A)^b \bmod p = (g^a)^b \bmod p = g^{ab} \bmod p$

} Correctness

Soundness objective of the attacker

start with same assumption here also

Integrity and Confidentiality ✓ ✓ → channel is assumed secure

without certificate → Anonymous D·H

fixed D·H

① public key of RSA
② Public key of D·H
③ certificate of RSA

⟹ Ephemeral D·H → Okay for completely insecure channel

Fixed D·H

→ session key remains same

→ time low

new session can not be operated using fixed

Ephemel D·H

Faster

Elliptic curve

HTTPS ✓✓

HTTPX

Green or lock ✓

Red or broken lock → ✗

attackers

given: $p, q, g, g^a \bmod p, g^b \bmod p$

find $g^{ab} \bmod p$

→ Diffie Hellman problem Vs Diffie Hellman protocol

Hard (or) not

a, b can not be calculated
from $g^a \bmod p$ and $g^b \bmod p$
given g, p, q

D·H security depends on DLP assumption
as long as DLP is Hard then D·H is secure

D·H can not be hard if DLP is not hard

D·H reduces to DLP

[any Hard problem can not be proved that it will remain hard]

Diffie Hellman Hardness assumption
DL Hardness assumption

D·H can not be hard if DLP is not hard

Hard ⟵ (DLP, DH Problem, Integer factorisation) ↗ library

KeyGen algo

① p ← large prime
② q ← large prime
③ $n \leftarrow p \times q$
④ $\phi \leftarrow (p-1)(q-1)$
⑤ $d \xleftarrow{R} Z_\phi^*$
⑥ $e \leftarrow d^{-1} \bmod \phi$

take help from library

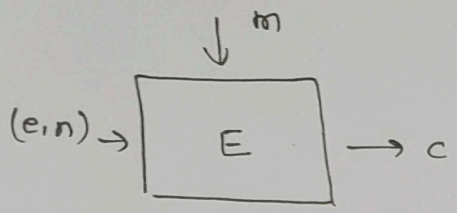
not n

(e, n) public key
(d, n) → private key

Keep loop { d is random element from Z_ϕ^*

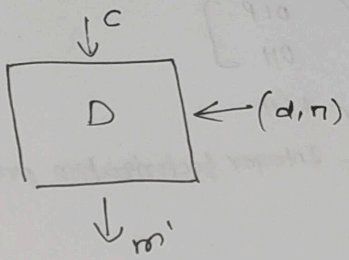
library ⟵ something b/w 1 and $\phi - 1$
and it should be coprime with ϕ

Encryption



1) $c = m^e \bmod n$

Decryption



$(m' = c^d \bmod n)$

correctness proof

$$m' = c^d \bmod n$$
$$= (m^e)^d \bmod n$$
$$= (m^{ed}) \bmod n$$
$$= m^1 \bmod n \quad [e = d^{-1}]$$
$$= m$$

Benny Madhvan

ed = 1

is wrt ϕ then why we are using wrt n why

Soundness of RSA

identify the attacker problem

→ to get m

algo is also known by default to attacker

given : e, n, c

find : m

because $\phi = (p-1)(q-1)$

using Euler's theorem we can say

RSA problem

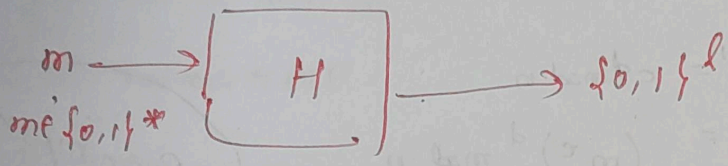
✓ # RSA is not secure if Integer factorisation is not hard

$p \rightarrow q \equiv \neg q \rightarrow \neg p$

RSA is hard because Integer factorisation is Hard ✗

{ [DLP, DH], [RSA, Integer factorization problem] }

Hash function



it should be deterministic

$$H \equiv \{0,1\}^* \longrightarrow \{0,1\}^l$$

Random Oracle model → deterministic algo