

be done insecurely, can be done securely if DDH assumption holds.

→ RSA Algorithm:

- Key generation: choose 2 large primes p & q (distinct) & choose e, d such that $ed \equiv 1 \pmod{(p-1)(q-1)}$

Public key $\langle pq, e \rangle$

Private key $\langle d, pq \rangle$

- Encryption: $m \in \{1, 2, \dots, N-1\}$

$$c = m^e \pmod{N}$$

- Decryption: $c \in \{1, 2, \dots, N-1\}$

$$m = c^d \pmod{N}$$

$$\text{Now } (m^e)^d \pmod{N} = m^{ed} \pmod{N}$$

$$\text{Note } m^{\phi(N)} \pmod{N} = 1$$

$$\text{Now } N = pq \Rightarrow \phi(N) = \phi(pq) = pq - p - q + 1 = (p-1)(q-1)$$

$$\therefore m^{(p-1)(q-1)} \pmod{N} = 1$$

$$\Rightarrow m^{ed} \pmod{N} = m^{ed - (p-1)(q-1)k} \pmod{N} = 1 \cdot m = m \quad \forall k$$

$$\Rightarrow m^{ed} \pmod{N} = m \pmod{N} = m$$

→ some possible attacks

- 1) if $e=3$ (or small) { min value of $e=3$ }

given $c, N \approx m$ ($m < \sqrt{N}$)

$$c = m^3 \pmod{N} \Rightarrow c = m$$

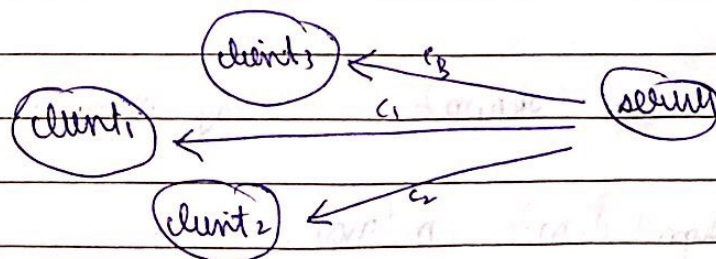
$$\Rightarrow m = c^{1/3}$$

2)

$e=3$

m is not

small



$$c_1 = m^3 \bmod N_1$$

$$c_2 = m^3 \bmod N_2$$

$$c_3 = m^3 \bmod N_3$$

Find x st $x = c_1 \bmod N_1$

$$x = c_2 \bmod N_2$$

$$x = c_3 \bmod N_3$$

$$x \in [0, N_1 N_2 N_3 - 1] \quad \text{CRT}$$

$$\text{Now } x = m^3$$

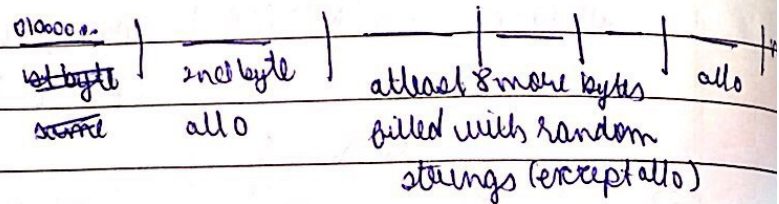
$$\Rightarrow m = x^{1/3}$$

→ PKCS V1.5 (does not have a provable security)

Key gen: same

Encryption: $c = m'^d \bmod N$

$m = k$ byte
no.

$m' =$ 

Decryption: get m'

Note: Pad is present because MSBs are easy to find in RSA.

→ RSA signatures:

$A \xrightarrow{m, G_A} B$ authenticate m

$$G_A = \text{sign}(sk_A, m)$$

$$\text{verify}(m, G_A, pk_A) \rightarrow Y/N$$

$$\text{sign}(d, m) = m^d \bmod N = G.$$

Verify $(m, G, \langle N, e \rangle) = \text{Y if } G^e \bmod N = m$

→ Attacks:

1) Don't know d

$$m = G^e \bmod N \quad G = \text{any random } G$$

2) Improved attack

$\langle m_1, G_1 \rangle$

$\langle m_2, G_2 \rangle$

New send $\langle m_1 m_2 \bmod N, G_1 G_2 \bmod N \rangle$

→ hash & sign paradigm (crn) (not proved)

$$G = [h(m)]^d \bmod N$$

verify: Y if $G^e \bmod N = h(m)$

→ How to authenticate the user?

Ask the user p to prove that he knows the secret key.

→ Bit commitment

binding $\rightarrow [b]$

blinding $\rightarrow b$ is secret

commit to a bit b & later reveal it.

• Commit phase:

Pick a random s .

Publish $\langle f(s), h(s) \oplus b \rangle$

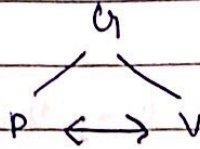
• Reveal phase:

Give b & s .

if $f(s)$ is same $\Rightarrow h$ not changed

if $h(s) \oplus b$ is same $\Rightarrow b$ not changed.

Problem: graph 3 colouring
Given a graph $G=(V,E)$ is it a 3 colourable.



code for prover (3 colours are RGB)

1. Enumerates the 3 colours.
2. Creates n locked boxes, each with the colour of the corresponding vertex.

$\boxed{c_1} \quad \boxed{c_2} \quad \dots \quad \boxed{c_n}$ sends them to V .

code for V

1. V picks one edge (i,j) at random & asks to reveal c_i & c_j .
if $c_i = c_j$ reject
else repeat / accept

Completeness: If G is 3 colourable, V always accepts.

Soundness: If G isn't 3 colourable, V rejects w.h.p.
prob of acceptance = $(\frac{1}{3})^k$

ques $y = g^x \text{ mod } p$ y is public including g & p .
give a $z \in \mathbb{Z}_p$ for x

choose a random no from \mathbb{Z}_p^*
 $P \rightarrow V : t = g^x \text{ mod } p$

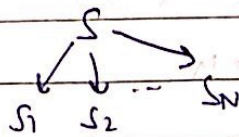
$v \rightarrow p$: some random c .

$p \rightarrow v$: $z = x + c$ $z = (x + c)$

v checks $g^u = y^{r \cdot t} \Rightarrow \text{accept}$

Replace c with $H(g, p, y)$

\rightarrow secret sharing



$$0 \leq t < n$$

Any $\geq (t+1)$ S 's can reconstruct S

Any $\leq t$ S 's have no information about S .

Shamir's secret sharing:

Define $S \in F$ (eg \mathbb{Z}_p prime p)

$$Q(x) = \sum_{j=0}^t a_j x^j$$

x_1, x_2, \dots, x_n be distinct public elements of F $S_i = Q(x_i)$

$Q(0) = x_0 = S$ $a_j, j > 0 = \text{random from } F$

general access structure:

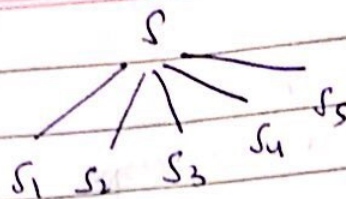
$$A \subseteq 2^{[1, \dots, n]}$$

All subsets $B \in A$ can access S

Others should not have no info about S .

$B \in A$ if $|B| \geq t+1$ else $B \notin A$

$$A = \{B \mid |B| \geq t+1\}$$



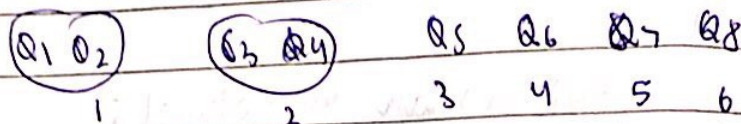
$$A = \{ \{1,2,3\}, \{3,4,5\}, \{2,3,4\} \}$$

eg

$$n=6$$

$$A = \{ \{1,2,3\}, \{3,4,5,6\}, \{1,3,4,5\} \}$$

degree $t=3$ ≈ 8 points



A = access structure.

$$f: \{0,1\}^n \rightarrow \{0,1\}$$

$$A_f = \{B \mid f(B) = 1\}$$

$$f(B) = 1 \quad \text{if } |B| \geq t+1$$

$$0 \quad \text{if } |B| \leq t$$

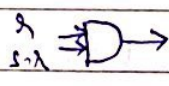
$f(B)$ is monotone:

→ if $f(B) = 1$ for a set, it will be 1 for all supersets.

→ can be built using only AND & OR.

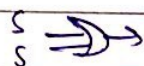
→ From AND & OR circuits for IA to secret sharing schemes for IA

eg



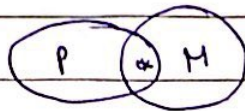
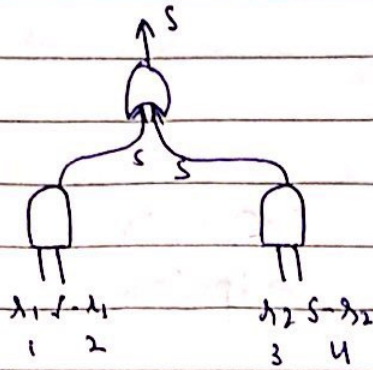
$$IA = \{ \{1,2,3\} \}$$

$$I\bar{A} = \{ \{1,3\}, \{2,3\}, \emptyset \}$$



$$IA = \{ \{1,3\}, \{2,3\}, \{1,2,3\} \}$$

$$I\bar{A} = \{ \emptyset \}$$



PM = polynomial AND-OR circuit

→ Oblivious Transfer (OT)



index i

$B = [b_1, b_2, \dots, b_n]$

output = b_i

shouldn't know i

shouldn't know $b_j \neq i$



$f(i, B) = B[i]$

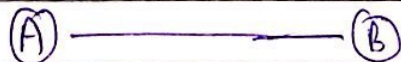
$f(x, y)$ efficient algo to compute f

if we can solve OT, we can ~~prove~~ solve any problem.

Proof

$h(i, B) = B[i]$

there can be explosion of DA is large.



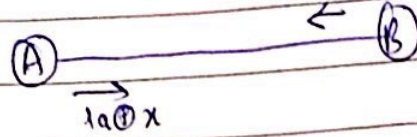
$x \in D_A$

$y \in D_B$

$[1 \dots n]$

$B = [f(1, y), f(2, y), \dots, f(n, y)]$

$f(x, y) = B[x] = h(x, B)$

XOR (x, y)AND (x, y) $x_b \oplus y$  $x_a, x_b \oplus y$ $x_a \oplus x, x_b$ Output z_a Output z_b

$$z: z_a \oplus z_b = f(x, y)$$

→ Secure AND

$$x_a \oplus x_b = x$$

$$y_a \oplus y_b = y$$

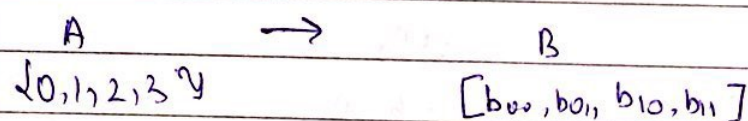
$$\text{Ans } z_a \oplus z_b = x \wedge y$$

$$= (x_a \oplus x_b) \wedge (y_a \oplus y_b)$$



x_A	y_A	z_a
0	0	$z_b \oplus (x_b \wedge y_b)$
0	1	$z_b \oplus (x_b \wedge \bar{y}_b)$
1	0	$z_b \oplus (\bar{x}_b \wedge y_b)$
1	1	$z_b \oplus (\bar{x}_b \wedge \bar{y}_b)$

can be computed by B.

Now A executes OT with $n=4$ to get z_A .

∴ If we have OT, we can do secure AND.

→ Secure XOR can be done with trusted 3rd party.

After doing secure AND & secure XOR, we can do any function ~~securely~~ secure 2 party communication.

→ Now how to do OT protocol:

Step 1 A sends a random array to B except its element.

$$A = [x_1, x_2, \dots, x_i, \dots, x_n]$$

all random

$$x_i = \text{Enc}_B(x_i) \quad \text{say RSA then } x_i = x_i^e \bmod N$$

Step 2 B decrypts the entire array A to obtain D.

$$D = [\text{Dec}(x_1), \text{Dec}(x_2), \dots, x_i, \dots, \text{Dec}(x_n)]$$

Add D & B

$$DB = [\forall j, \text{Dec}(x_j) \oplus b_j]$$

Send DB to A.

Step 3. A obtains b_i as

$$b_i = DB[i] \oplus x_i$$