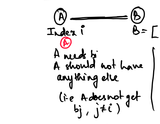
Using
i) PPTM adversary.
ii) Negligible Error Probability.

We achieved
→ Symmetric Key Cipher Q) What can be achieved
→ Data Integrity A) Everything
→ Public Key Crypto
→ Signatures

Oblivious Transfer.



OT Protocol

Step 1:- A chooses a random array

$r = [r_1, r_2, \ldots\ldots r_n]$

encrypts only ith bit with its PK (Bit he wants to receive from B)

$r' = [r_1, r_2, \ldots\ldots Enc_{PK_B}(r_i), \ldots\ldots r_n]$

sends to B

Step 2:-

receives r'
can't distinguish from completely random string.

does not know i

Decrypts the entire string

$\sigma = [\sigma_1, \sigma_2 \ldots\ldots \sigma_n]$

$Dec_{SK_B}(r_i)$ Note:- $\sigma_i = \beta_i$

And sends z

$z = [\sigma_1 \oplus b_1, \sigma_2 \oplus b_2 \ldots\ldots \sigma_n \oplus b_n]$

Step 3:-

A receives z and retrieves b_i as

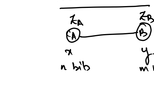
$b_i = z[i] \oplus \beta_i$

$= \beta_i \oplus b_i \oplus \beta_i$

$= b_i$ //

This is a semi-honest scheme, you rely on A to delete the array r after he calculates b_i

General Scheme



B does not know x
A does not know y
need to calculate $f(x,y)$

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

Say there are Z_A & Z_B generated by A & B such that

$Z_A \oplus Z_B = f$

Scheme

Step 1:-) A chooses Z_A from $\{0,1\}^k$

Step 2:-) $\forall\, y_i$ A calculates $f(x,y_i)$ sends $f(x,y_i) \oplus Z_A = Z_{Bi}$ (G_2^m size array)

Step 3:-) B knows the correct y_i, ∴ knows correct index i, ∴ gets Z_{Bi}

Now, both can reveal Z_A & Z_B and calculate $Z_A \oplus Z_B = f(x,y)$ without having to reveal x, y

Issue is that one need to calculate $f(x,y_i)$ for 2^m diff. y_i's

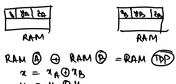
Even if f is polynomial size circuit we still need Exponential time

Solution

Every polynomial size circuit can be simulated using
i) AND gate.
ii) XOR gate

Have a TDP (Trusted Third Party)

Private 2 Party Computation



RAM (A) ⊕ RAM (B) = RAM (TDP)

$x = x_A \oplus x_B$

$y = y_A \oplus y_B$

$z = z_A \oplus z_B$

Instruction Set Architecture.

i) Calculating XOR privately.

$z = x \oplus y$

$= (x_A \oplus x_B) \oplus (y_A \oplus y_B)$

$= (x_A \oplus y_A) \oplus (x_B \oplus y_B)$

$= z_A \oplus z_B$

∴ XOR is simple, just store $x_A \oplus y_A$ individually in some memory & $z_A \oplus y_B$

ii) Calculating AND privately.

Need to use Oblivious transfer → TDP calculates (3)

$z = (x \wedge y)$

$= (x_A \oplus x_B) \wedge (y_A \oplus y_B)$ not commutative

$Z_A \leftarrow \{0,1\}^k$ randomly.

→ A calculates an array of size (4)

$[Z_A \oplus (\bar{x}_A \bar{y}_A),\ Z_A \oplus (\bar{x}_A y_A),\ Z_A \oplus (x_A \bar{y}_A),\ Z_A \oplus (x_A y_A)]$

(x_B, y_B) ↑ (0,0) ↑ (0,1) ↑ (1,0) ↑ (1,1)

→ B uses his values of (x_B, y_B) to obtain the correct index of the array (Z_B)

→ A & B store z_A & z_B in same location, TDP gets z in that location.

Example.

$x_B, y_B = (1,0)$

$z = x \wedge y$

$= (x_A \oplus x_B) \wedge (y_A \oplus y_B)$

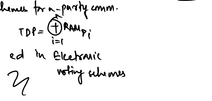
$= (x_A \oplus 1) \wedge (y_A \oplus 0)$

$= \bar{x}_A y_A$

B retrieves 3rd index from array.

$= z_A \oplus \bar{x}_A y_A$

$z_A \oplus \bar{x}_A y_A = z$



Similar Schemes for n-party comm.

↳ TDP = $\bigoplus_{i=1}^{n} RAM_{P_i}$

be used in Electronic voting schemes