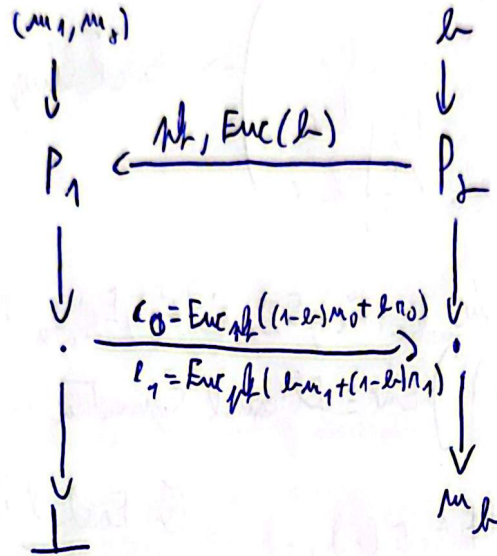


# Privacy :

TP.4.0

TP4:

Q1)



$$\begin{aligned}
 r_0, r_1 &\leftarrow \text{rand}(2^{\lambda})p_1 \\
 m_0, m_1 &\leftarrow 2^{\lambda}p_1 \\
 r &= \text{Enc}_{pk}(b) \\
 y &= \text{Enc}_{pk}(1-b) \\
 &= \text{Enc}_{pk}(1) \cdot r^{-1} \% p \\
 c_0 &= y^{m_0} \cdot r^{r_0} \% p \\
 r_1 &= r^{m_1} \cdot y^{r_1} \% p
 \end{aligned}$$

$$\leftarrow \text{Enc}_{pk}(b)$$

$$r_0, r_1$$

$$\begin{aligned}
 p_2 \quad b &\in \{0, 1\} \\
 r_k &\leftarrow 2^{\lambda} 1 \wedge k = 1^{r_k} \% p
 \end{aligned}$$

$$\text{if } b=0 : c=r_0 \text{ else } : c=r_1$$

$$\text{Dec}_{pk}(c) = \log_g^* (g^{m_c}) = m_c$$

Check pattern( $m_c$ )

~~Example~~ Feasible because  $m_c$  is small enough!!

Q2)

CARBLING PHASE

1)

| a | b | c |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

MAP  
(=)

| a       | b       | c       |
|---------|---------|---------|
| $k_a^0$ | $k_b^0$ | $k_c^0$ |
| $k_a^0$ | $k_b^1$ | $k_c^0$ |
| $k_a^1$ | $k_b^0$ | $k_c^0$ |
| $k_a^1$ | $k_b^1$ | $k_c^1$ |

} → Carbling Keys

Carbled Table = shuffle  $([E_{k_a^0, k_b^0}(k_c^0), E_{k_a^0, k_b^1}(k_c^0), E_{k_a^1, k_b^0}(k_c^0), E_{k_a^1, k_b^1}(k_c^1)])$

$(E_{A,B}(C) = Enc_A(Enc_B(C || 0^M))) \wedge Enc \equiv El$  (usual Symmetric Encryption)

2)

DATA TRANSFER PHASE(S)

Each bits generation has its own collection  $(k_a^0, k_b^1, k_a^0, k_b^1, k_c^0, k_c^1)!$

Stream bits  $a$   $P_0$  (Carbler)

$P_1$  (Executor)

For each bit  $i$  \*

$a_i \in a$ :

map:  $i \rightarrow (k_a^0, k_b^1, k_a^0, k_b^1, k_c^0, k_c^1)$

compute carbled table  $T_i$

map:  $a_i \rightarrow k_a^{a_i}$

Compute challenges  $r_0, r_1$   
for  $0 \neq$  where  $m_0 = k_a^0$   
 $\wedge m_1 = k_b^1$ .

$k_a^{a_i}, T_i$

Stream bits  $b, c, k$   
Take next bit  $b_i$

$k_b^{b_i}, Enc(b_i)$

With oblivious Transfer to get safely  $k_b^{b_i}$

$r_0, r_1$

$Dec_{k_b^{b_i}}(k_b^{b_i}) = \log_g(g^{k_b^{b_i}}) = k_b^{b_i}$

$Dec_{k_a^{a_i}, k_b^{b_i}}(T_i) =$

$[m, m, k_c^{a_i \wedge b_i}, m]$  (order could be different.)

Get valid decryption  $k_c^{a_i \wedge b_i}$  with  $m$  as ending zeros, all others are invalid.

map:  $k_c^{a_i \wedge b_i} \rightarrow r_i = a_i \wedge b_i$

$k_c^{a_i \wedge b_i}$



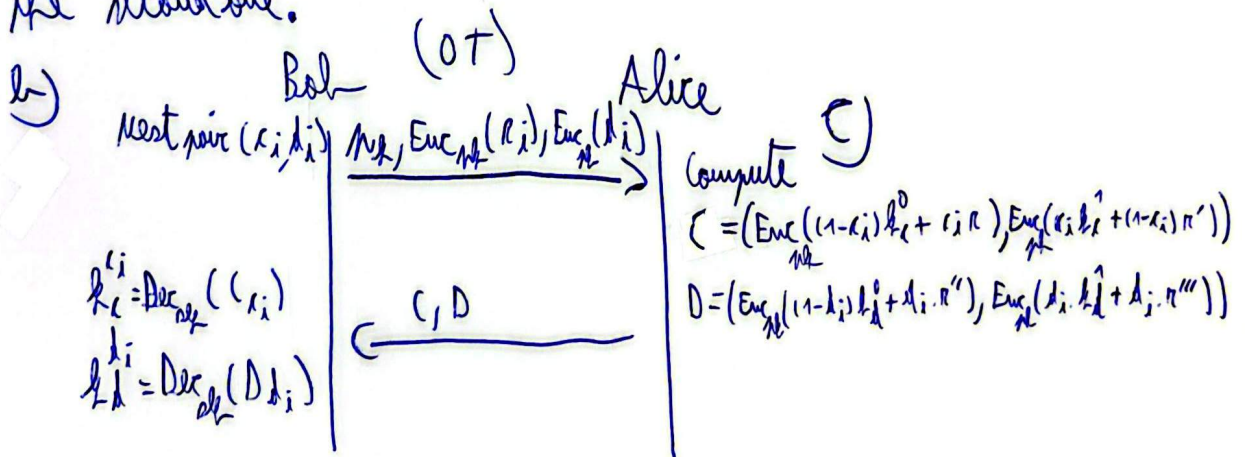
3)

| A | B | C | D | A ⊕ B | A ⊕ C | A ⊕ D | B ⊕ C | B ⊕ D | C ⊕ D | E F T P. 7.2 |
|---|---|---|---|-------|-------|-------|-------|-------|-------|--------------|
| 0 | 0 | 0 | 0 | 0     | 0     | 0     | 0     | 0     | 0     |              |
| 0 | 0 | 0 | 1 | 0     | 0     | 1     | 0     | 1     | 1     |              |
| 0 | 0 | 1 | 0 | 0     | 1     | 0     | 1     | 0     | 1     |              |
| 0 | 0 | 1 | 1 | 0     | 1     | 1     | 1     | 1     | 0     |              |
| 0 | 1 | 0 | 0 | 1     | 0     | 0     | 1     | 0     | 0     |              |
| 0 | 1 | 0 | 1 | 1     | 0     | 1     | 1     | 1     | 1     |              |
| 0 | 1 | 1 | 0 | 1     | 1     | 0     | 0     | 1     | 0     |              |
| 0 | 1 | 1 | 1 | 1     | 1     | 1     | 0     | 0     | 1     |              |
| 1 | 0 | 0 | 0 | 1     | 1     | 1     | 0     | 0     | 1     |              |
| 1 | 0 | 0 | 1 | 1     | 1     | 0     | 0     | 1     | 0     |              |
| 1 | 0 | 1 | 0 | 1     | 0     | 1     | 1     | 1     | 1     |              |
| 1 | 0 | 1 | 1 | 1     | 0     | 0     | 1     | 0     | 0     |              |
| 1 | 1 | 0 | 0 | 0     | 1     | 1     | 1     | 0     | 1     |              |
| 1 | 1 | 0 | 1 | 0     | 1     | 0     | 1     | 1     | 0     |              |
| 1 | 1 | 1 | 0 | 0     | 0     | 1     | 0     | 0     | 1     |              |
| 1 | 1 | 1 | 1 | 0     | 0     | 0     | 0     | 0     | 0     |              |

c) Alice for each pair  $(a_i, b_i)$  of her stream input create a mapping collection  $(k_a^0, k_a^1, k_b^0, k_b^1, k_c^0, k_c^1, k_d^0, k_d^1, k_e^0, k_e^1, k_f^0, k_f^1)$  where each  $k_x^B$  means label for variable  $x$  equaling boolean  $B$ .

There are 2 garbled tables containing  $2^4 = 16$  entries resulting from the shuffle of the list of all possible combination  $(a_i, b_i, c_i, d_i)$  and each entry is encoded as

$\text{Enc}_{k_a^0}(\text{Enc}_{k_b^1}(\text{Enc}_{k_c^1}(\text{Enc}_{k_d^1}(k_{a,b,c,d}^{a_i, b_i, c_i, d_i} \parallel 0^n)))$  where  $n$  is a security parameter and  $n = l$  for the first garbled list and  $f$  for the second one.



d)  $DT_{Ei} = \text{Dec}_{k_a^0}(\text{Dec}_{k_b^1}(\text{Dec}_{k_c^1}(\text{Dec}_{k_d^1}(T_{Ei}))))$  Then gets the only decrypted value valid with  $n$  ending zeros.

Garbled table for  $E$ . This value is  $k_e^0$ .

Same logic for extracting  $k_f^0$  with  $T_{Fi}$ .

4) ~~When integers are > 64 bits we need big ints, virtual integers using several registers. It impacts drastically the performance.~~

~~When exponents get huge and then generate very big cypher texts that complexity the workload and overload the network.~~

Lower classical keys can be broken now using brute force (ex for 80 bits keys).  
On the other hand 128 bits key are robust enough ( $2^{81474}$  billions times harder to brute force).