Privacy Engineering (70018)

MPC 2 - Questions

2.1 Compute the garbled tables for the AND-XOR-OR circuit in the slides with p-bit values `p[1]=0, p[2]=1, p[3]=0, p[4]=1, p[5]=1, p[6]=1, p[7]=0`

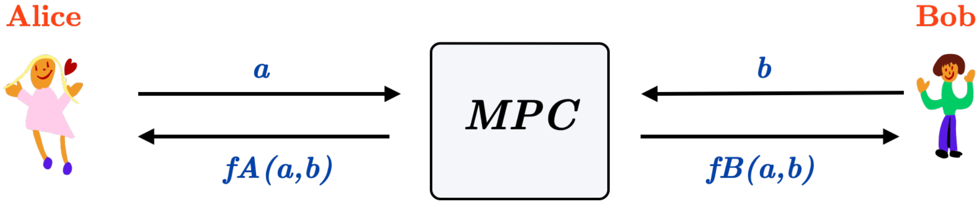
2.2 Exam 2018. Produce the garbled table for a NAND gate with input wires *w1* and *w2* and output wire *w3* where the *p*-bit is 1 for all three wires. Clearly show your solution on a diagram for the NAND gate, labelling the wires with their key/index-bit pairs as well as showing the encrypted table entries.

2.3 Exam 2019. Construct the garbled table for an XOR gate with input wires *w1* and *w2* and output wire *w3* with *p*-bits 1, 1, 1 for wires *w1*, *w2* and *w3* respectively. Clearly show your solution using a diagram for the XOR gate, labelling the wires with their keys and permutation bits as well as listing the garbled table entries.

Assume Alice constructs a garbled circuit with one XOR gate is constructed as in part 1a. Wire *w1* is for Alice's input. Wire *w2* is for Bob's input. Show the steps that Bob takes to evaluate the circuit when Alice's input bit is 1 and Bob's input bit is 0.

2.4 <u>Optional</u>. Derive an un-encrypted logic circuit for the millionaire's problem for 2-bit Alice and 2-bit Bob values. Output should be 1 if Bob > Alice, otherwise 0.

2.5 Devise an approach for 2-party MPC protocol where Bob computes separate functions:



Here Alice learns $fA(a, b)$ without Bob learning a or $fA(a, b)$ while Bob learns $fB(a, b)$ without Alice learning b or $fB(a, b)$.

2.6 Exam 2015. Consider the following 1-from-n oblivious transfer protocol in an honest-but-curious (semi-honest) model.

1. Alice generates n random public-private key pairs

   $$(pub_1, priv_1), \dots (pub_n, priv_n)$$

   Alice sends the public keys $pub_1, \dots, pub_n$ to Bob.

2. Bob generates n random symmetric keys $k_1, \dots, k_n$ and computes

   $$G_b = E_{pub_b}(k_b) \quad \text{and} \quad G_z = k_z \text{ for all } z \in \{1..n\} \text{ and } z \neq b$$

Bob sends G_1 to G_n to Alice. b is Bob's selection ∈ {1..n}

3. Alice computes

$$H_z = D_{priv_z}(G_z), \quad C_z = E_{H_z}(M_z) \quad \text{for all } z \in \{1..n\}$$

Alice sends C_1 to C_n to Bob

4. Bob computes $M_b = D_{k_b}(C_b)$

For this protocol:

i) Show that Bob's output equals M_b.

ii) Explain why Alice learns nothing about b. What assumptions do you have to make about the two cryptosystems for this to be true?

iii) Explain why Bob learns nothing about M_z for $z \neq b$. What assumptions do you have to make about the two cryptosystems for this to be true?

iv) If Alice were dishonest, is there anything she could do to learn b? If so, describe how. If not, explain why not.

v) If Bob were dishonest, is there anything he could do to learn messages other than M_b? If so, describe how. If not, explain why not.

2.7 Exam 2016. Consider the following *1*-from-*2* oblivious transfer protocol based on the well-known Diffie-Hellman key-exchange protocol in an honest-but-curious setting.

1. Alice generates a random number a (from $\mathbb{Z}p$). Similarly, Bob generates a random number b. Bob's message selection bit is m.

2. Alice sends $A = g^a$ to Bob. g is a suitable generator for the group.

3. If $m=0$ Bob sends $B=g^b$ to Alice.

   If $m=1$ Bob sends $B=Ag^b$ to Alice.

4. Alice computes: $k_0 = Hash(B^a)$ $C_0 = E_{k_0}(M_0)$

   $k_1 = Hash((B/A)^a)$, $C_1 = E_{k_1}(M_1)$

   Alice sends C_0 and C_1 to Bob.

5. Bob computes $k = Hash(A^b)$, $M_m = D_k(C_m)$,

For this protocol:

i) Explain why Bob's output equals M_m.

ii) Explain why Alice learns nothing about m and why Bob learns nothing about M_z for $z \neq m$. What assumptions do you have to make about the two cryptosystems for this to be true?

iii) Explain what, if any, issues arise if Alice sets a to 0. What if Bob sets b to 0 (with Alice generating a random number a as normal)?

2.8 Exam 2018. Consider the following *1*-from-*2* oblivious transfer protocol that uses a trusted third-party Trent in an *honest-but-curious* setting. Alice's messages M_0 and M_1 are binary values of length k. Bob's message selection bit is b.

1.	Trent → Alice: R_0, R_1	Random binary values each of length k
2.	Trent → Bob: t, R_t	Random bit t
3.	Bob → Alice: e	$e = t \oplus b$
4.	Alice → Bob: C_0, C_1	$C_0 = M_0 \oplus R_e$, $C_1 = M_1 \oplus R_{1-e}$
5.	Bob: M_b	$M_b = C_b \oplus R_t$

For this protocol:

i) Show the working for M_0=1101, M_1=0100, b=1, t=0, R_0=0101, R_1=0011.

ii) Explain how the protocol satisfies the properties of an oblivious transfer. Could Alice or Bob learn anything if they were malicious?

iii) Adapt the protocol to a 1-from-n oblivious transfer. Assume that n is a power of 2, i.e. 2, 4, 8, ...

2.9 Exam 2019. Consider the following *1*-from-*2* oblivious transfer protocol based on the well-known ElGamal encryption scheme in an *honest-but-curious* setting. All operations and random numbers are for a suitable group $\mathbb{Z}/q\mathbb{Z}$ of prime order q and generator g.

Alice's messages are M_0 and M_1 of length n. Bob's message selection bit is b.

1. Alice generates random numbers x and k.

   Alice sends x to Bob.

2. Bob generates a random number y and computes $H_b = g^y$ and $H_{1-b} = x/H_b$.

   Bob sends H_0 and H_1 to Alice.

3. Alice computes D=g^k, $C_0 = M_0 \oplus Hash(H_0{}^k)$, $C_1 = M_1 \oplus Hash(H_1{}^k)$
   Hash is a cryptographic hash function that produces values of length n.

   Alice sends D, C_0 and C_1 to Bob.

4. Bob computes $M_b = C_b \oplus Hash(D^y)$

For this protocol:

i) Explain why Bob's output equals M_b.

ii) Explain why Alice learns nothing about b and why Bob learns nothing about M_z for $z \neq b$.