ETH Zurich, Department of Computer Science
SS 2019

Prof. Ueli Maurer
Dr. Martin Hirt
Chen-Da Liu Zhang

Cryptographic Protocols
Exercise 7

7.1 Permuted Truth Tables

In their protocol, which we discussed in the lecture, Brassard, Chaum, and Crépeau use "permuted" truth tables of binary logical operations.

x	y	x ∧ y
0	0	0
0	1	0
1	0	0
1	1	1

truth table

x	y	x ∧ y
1	0	0
1	1	1
0	1	0
0	0	0

"permuted" truth table

In this exercise we consider an alternative way of processing $\wedge$-gates:

a) Assume that a commitment scheme of type B is given along with a protocol that allows to prove in zero-knowledge that two blobs are commitments to equal values. Let c_1, c_2, and c_3 be blobs for the bits b_1, b_2, and b_3, respectively. Construct a zero-knowledge protocol which allows Peggy to convince Vic that $b_3 = b_1 \wedge b_2$. Show that your protocol is complete, sound, and zero-knowledge.

Hint: Use an approach based on "permuted" truth tables.

b) Show how Peggy can use the above construction to prove for an arbitrary circuit that she knows an input that evaluates to a given output.

c) What is the difference between the process from **b)** and the one described in the BCC protocol?

7.2 Protocols and Specifications

Parties P_1 and P_2 hold input bits x_1 and x_2, respectively. They want that P_2 learns the AND of their inputs.

Specification 1

```
*$P_1$ (resp. $P_2$) holds input bit $x_1$ (resp. $x_2$).*
1: $P_1$ (resp. $P_2$) sends $x_1$ (resp. $x_2$) to the trusted party.
2: The trusted party sends $y = x_1$ to $P_2$.
3: $P_2$ outputs $y$.
```

Specification 2

```
*$P_1$ (resp. $P_2$) holds input bit $x_1$ (resp. $x_2$).*
1: $P_1$ (resp. $P_2$) sends $x_1$ (resp. $x_2$) to the trusted party.
2: The trusted party sends $y = x_1 \wedge x_2$ to $P_2$.
3: $P_2$ outputs $y$.
```

Protocol 3

```
*$P_1$ holds input bit $x_1$, $P_2$ holds input bit $x_2$.*
1: $P_1$ sends $x_1$ to $P_2$.
2: $P_2$ computes $y = x_1 \wedge x_2$.
3: $P_2$ outputs $y$.
```

a) Does Protocol 3 satisfy Specification 1 in the case where both parties are honest? What about Specification 2?

b) Does Protocol 3 satisfy Specification 2 when the adversary passively corrupts P_2? What if the adversary actively corrupts P_2?

Now consider three parties P_1, P_2 and P_3 with input bits x_1, x_2 and x_3, respectively. They want that P_1 and P_3 learn the AND of the three inputs.

Specification 4

```
*$P_1$ (resp. $P_2$, $P_3$) has input bit $x_1$ (resp. $x_2$,$x_3$)*
1: Each party $P_i$ sends $x_i$ to the trusted party.
2: The trusted party sends $y = x_1 \wedge x_2 \wedge x_3$ to $P_1$ and $P_3$.
3: $P_1$ and $P_3$ output $y$.
```

Protocol 5

```
*$P_1$ (resp. $P_2$, $P_3$) has input bit $x_1$ (resp. $x_2$,$x_3$)*
1: $P_1$ sends $x_1$ to $P_2$.
2: $P_2$ sends $y_2 = x_1 \wedge x_2$ to $P_3$.
3: $P_3$ sends $y_3 = y_2 \wedge x_3$ to $P_1$.
4: $P_1$ and $P_3$ output $y_3$.
```

c) Does Protocol 4 satisfy Specification 5 when the adversary passively corrupts P_1 and P_2? What about P_1 and P_3? Is there a subset of players the adversary can passively corrupt so that the protocol is secure? For the same sets of corrupted players, analyze the protocol when the adversary is active.