Aalto University

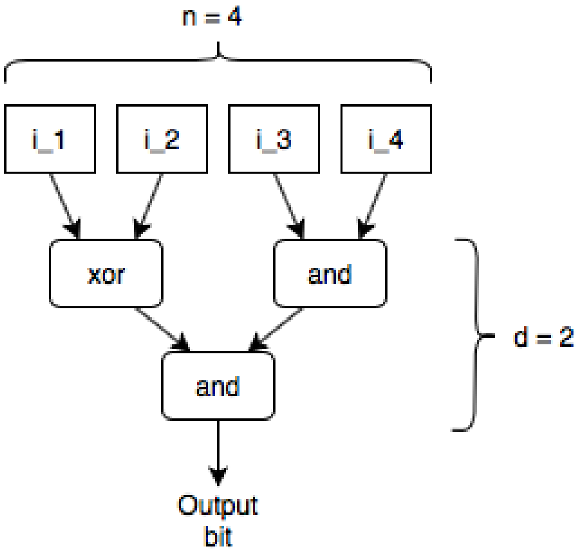
MS-E1687 Advanced Topics in Cryptography

Exercise Sheet 8

Deadline: 23:55, March 28, 2019, via MyCourses. A reasonable answer to 1 exercise corresponds to 1 point, reasonable answers to 2 exercises correspond to 2 points. This sheet gives at most 2 points. The exercise sheets are intended to support your learning and encourage experiments and free thinking. Grading should not stand in the way of exploration. In practice, that means that points are given for reasonable answers and that the course is only pass/fail and does not have grades. If you hand in your solutions by Thursday 21, feedback will be given a week earlier than handing in the solutions by March 28.

Exercise 22: (Learning) Let d be a constant. We define a concept class F as follows: For $n \in \mathbb{N}$, F_n contains all circuits (using xor and and gates with fan-in 2) in n variables of constant depth d. Show that with high probability, for all $n \in \mathbb{N}$ and for all $f \in F_n$, the function f can be PAC-learned with n^{2^d} samples in polynomial-time.
Hint: We recommend to use polynomial interpolation.

Solution: (Sketch) The xor gates can be thought of as addition modulo 2 and and gates can be thought of as multiplication modulo 2. Hence the output of the circuit is the same as the result of a certain polynomial in $\mathbb{Z}_2$. Since the depth of the circuit is d, the polynomial is at most of degree 2^d.
See for example the circuit in the figure below.



That circuit corresponds to the polynomial

$$(i_1 + i_2) \cdot (i_3 \cdot i_4) = i_1 \cdot i_3 \cdot i_4 + i_2 \cdot i_3 \cdot i_4 \mod 2 \tag{1}$$

whose degree is $3 < 2^d = 2^2$.
Recall from linear algebra that if you have an unknown polynomial of degree m and you get m values of the poly with the respective different inputs, then you can form m linear equations. If the equations are linearly independent, you can find the polynomial using Gaussian elimination (in polynomial time).
Now the learner cannot choose the inputs, so they might receive the same input – output pair multiple times. However, if they try long enough, it is likely that they get enough linearly independent equations and can find the polynomial. Once the learner has a poly, it is easy to find a circuit that matches the poly and this circuit is a correct answer.

Exercise 23: (Zero-Knowledge) Prove the correctness, soundness and zero-knowledge property of the graph-isomorphism protocol given in the lecture.

Solution:

Correctness If the graph selected by the *Verifier*, b, is the graph permuted by the *Prover*, d, then ψ must transform $G_b(= G_d)$ into G with 100% accuracy. If the graph selected by the *Verifier* is not the graph permuted by the *Prover*, $b \neq d$, then $\psi \circ \phi$ must transform G_b into G as $\psi(G_b) = \pi(\phi(G_d)) = \pi(G_b) = G$.

Soundness If the two input graphs are not isomorphic (under ϕ) then the *Prover* cannot reliably provide a transformation ψ to transfrom G_b to G to the *Verifier*. If $d = b$ then the *Prover* can convince the *Verifier* that it knows the transformation (as it just provides the transformation it applied). However, if $d \neq b$ the *Prover* cannot convice the *Verifier* because G_0 and G_1 are not isomorphic under ϕ.
As this only allows the *Verifier* to determine if the *Prover* is lying $\frac{1}{2}$ of the time the *Verifier* has to play the game two times to achieve the $\geq \frac{1}{4}$ bound.

Zero-knowledge The *Verifier* does not learn anything from ψ as it is just a function that transfroms the particular graph G_d into G and does not generalize to other graphs (not even, critically, G_b).

Exercise 24: (Popular Science: Zero-Knowledge) A prover is given a deck of green cards and a deck of red cards, and assume that the prover is able to distinguish red cards from green cards. How can the prover use their ability to distinguish green cards from red cards to prove to the verifier that a particular pair of cards are of *different* color? We here assume that the verifier is not able to distinguish red cards from green cards.

Solution: The *Prover* gives two cards to the *Verifier* of different color. The *Verifier* takes one card in each hand and shows which card is in which hand to the *Prover*. The *Verifier* then hides the cards from the *Prover* (e.g. by holding their hands behind their back) and either does or does not switch the cards. The *Verifier* than presents the cards to the *Prover*, who has to tell if the *Verifier* switched the cards or not.

Correctness This protocol is correct as a *Prover* that can tell the two cards apart will be able to provide the *Verifier* with two different colored cards and can henceforth tell if the *Verifier* switched the cards (regardless of repititions). The *Verifier* must also be convinced that the *Prover* can distinguish (any) two cards as they cannot see if the *Verifier* switches the cards.

Soundness If the *Prover* cannot tell the difference between the two cards they cannot reliably tell if the *Verifier* switched the cards or not. By repeating the protocol a sufficient certainty threshold can be obtained.

Zero-knowledge Apart from the fact that two particular cards are different, the *Verifier* does not learn anything about the cards. The *Prover* never mentions their color, so either card provided to the *Verifier* can be red or green.

Exercise 25: (Popular Science: Zero-Knowledge) A prover is given a deck of green cards and a deck of red cards, and assume that the prover is able to distinguish red cards from green cards. How can the prover use their ability to distinguish green cards from red cards to prove to the verifier that a particular pair of cards are of *the same* color? We here assume that the verifier is not able to distinguish red cards from green cards.

Solution: The *Prover* gives te *Verifier* three cards, two of the same color (a and b) and one of the other color (c). Now the *Prover* tells the *Verifier* that they can distinguish between a and c as well as b and c (using the previous proof).

Correctness This protocol is correct as a *Prover* that can tell the difference between the two colors can tell that a is not equal to c and that b is not equal to c, and since there are only two colors available and c uses one of them, a and b must be the same color.

Soundness If the *Prover* cannot tell the difference between the two cards they cannot manage to convince the *Verifier* they can tell the difference between even a and c, and therefor not that they can tell two cards have the same color.

Zero-knowledge The *Verifier* learns nothing about the actual color of any specific card as the *Prover* never mentioned a specific card.