ETH Zurich, Department of Computer Science
SS 2019

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

Cryptographic Protocols
Exercise 1

1.1 Padlocks

a) Consider two combination padlocks (e.g., with three wheels with 10 positions each), which can only be opened by someone who knows the corresponding secret combinations. Vic does not know the combinations. Peggy claims that she knows the combination for one of the padlocks and would like to convince Vic of this fact while Vic should learn neither the combination nor for which padlock Peggy knows the combination. Describe a protocol that meets Peggy's requirements. Is your protocol complete and sound? Is the above task a proof of a statement or a proof of knowledge?

b) Consider 100 combination padlocks, where Vic knows the combinations for all of them. Give a protocol that allows Peggy to prove to Vic that she knows the combination for *at least one* of the padlocks (without revealing which one).

c) Consider 7 padlocks, where Vic knows the combinations for all of them. Peggy now claims that she knows the combinations to open *at least* two of them. Describe a protocol that allows Peggy to prove her claim to Vic without revealing the padlocks she knows.

HINT: Use the above proof several times for different subsets of the 7 padlocks.

1.2 Graph (Non-)Isomorphism

In the lecture we saw different interactive proofs between a prover Peggy and a verifier Vic. We saw a protocol for graph isomorphism, and a protocol for graph non-isomorphism (GNI), i.e., for proving that two graphs $\mathcal{G}_0$ and $\mathcal{G}_1$ are isomorphic or not, respectively.

a) Argue why the GNI protocol is not zero-knowledge.

b) A protocol is said to be honest-verifier zero-knowledge if it is zero-knowledge in the case where the verifier follows the protocol. Is the GNI protocol honest-verifier zero-knowledge?

c) Assume that there are three graphs. Construct a honest-verifier zero-knowledge protocol that allows Peggy to prove that not all three graphs are isomorphic.

1.3 Square Roots and Factoring

a) For

- $a = 2$, $n = 7$,
- $a = 25$, $n = 29$, and
- $a = 11$, $n = 35$,

compute all square roots r modulo n of a (if any exist). How many square roots modulo n can a have? What about $n = 105$, $a = 4$?

b) Let $m = pq$ be an RSA modulus. We show that computer square roots is as hard as factoring. Consider an algorithm A that given a quadratic residue a chosen uniformly at random, returns a square root r modulo m of a with probability $\alpha \in (0, 1]$. Provide an algorithm B that uses A to factor m. What is the success probability of B?