ETH Zurich, Department of Computer Science
SS 2017

Dr. Martin Hirt
Chen-Da Liu Zhang

Cryptographic Protocols
Notes 5

Scribe: Sandro Coretti (modified by Chen-Da Liu Zhang)

About the notes: These notes serve as written reference for the topics not covered by the papers that are handed out during the lecture. The material contained therein is thus a *strict* subset of what is relevant for the final exam.

This week, the notes treat commitment schemes and two examples thereof. Moreover, they discuss a zero-knowledge protocol for proving knowledge of a Hamiltonian cycle in a graph.

5.1 Commitment Schemes

A *commitment scheme* is an interactive protocol between a prover P and a verifier V (both probabilistic) and consists of two phases Commit and Open. The input to P in the Commit phase is some value x from some input space $\mathcal{X}$, (V has no input), and the output of V in the Open phase is either some $x' \in \mathcal{X}$ (in which case V is said to *accept*) or $\perp$ (in which case V is said to *reject*).

Informally, commitment schemes are required to satisfy the following requirements:

- Correctness: When both P and V follow the protocol, V always (i.e., with probability 1) outputs P's input x.
- Hiding: After the Commit phase, V has no information about P's input x.
- Binding: After the Commit phase, there exists only one value x that will be accepted by V in the Open phase.

One can show that any commitment scheme can be transformed into one in which the Open phase is non-interactive (which the reader should do as an exercise).

In this course, we will only treat commitment schemes in which the Commit phase is non-interactive as well. Thus, a commitment scheme can be characterized by a function $C : \mathcal{X} \times \mathcal{R} \to \mathcal{B}$ that maps a value $x \in \mathcal{X}$ and a randomness string r from some randomness space $\mathcal{R}$ to a so-called *blob* $b = C(x, r)$ in some blob space $\mathcal{B}$. The Open phase simply consists of the prover's sending (x, r) to the verifier, who checks that $C(x, r) = b$.

The Hiding and Binding properties can be either perfect or computational, i.e., against unbounded or computationally bounded adversaries, respectively. A commitment scheme is called *Type H* if it is perfectly hiding and *Type B* if it is perfectly binding. One can show that no commitment scheme can be of both Type H and Type B (cf. Exercise 6.2**a)**).

5.2 Hamiltonian Cycles

In this section we discuss a zero-knowledge proof for the statement that a particular graph contains a Hamiltonian cycle.[1] The protocol is also a proof of knowledge.

In the $(n \times n)$ adjacency matrix of $\mathcal{G}$, a Hamiltonian cycle corresponds to a constellation of n ones (the n edges of the cycle), where each row and each column contains exactly one 1. (This condition is necessary, but not sufficient. The reader may think what additional constraint needs to be satisfied as an exercise).

A protocol for the Hamiltonian cycle problem (HC) is of particular interest since HC is **NP**-complete. That is, every instance of an arbitrary decision problem $L \in \mathbf{NP}$ can be mapped to a graph $\mathcal{G}$ such that $\mathcal{G}$ contains a Hamiltonian cycle if and only if the instance is in L. Thus, a zero-knowledge proof for HC can be used to obtain zero-knowledge proofs for arbitrary **NP** problems. However, such proofs would be prohibitively inefficient (albeit polynomial) due to the overhead incurred by the corresponding reduction. In [BCC88], the authors provide a more efficient protocol for proving general **NP** statements.

The protocol as described here uses idealized bit commitments, i.e., it assumes that Peggy and Vic have access to an ideal commitment service. This simplifies the presentation and the proof, which are much more involved if the idealized service is replaced by actual commitments.

A single round of the following protocol proceeds as follows:

1. Peggy produces a graph $\mathcal{H} = \sigma\mathcal{G}\sigma^{-1}$ from $\mathcal{G}$ using a random permutation σ. She then commits to the adjacency matrix of $\mathcal{H}$.
2. Vic chooses a random challenge bit c. If $c = 0$, Peggy opens the commitments to all entries of the adjacency matrix and provides σ to Vic. If $c = 1$, Peggy only opens the n entries in the adjacency matrix that correspond to the Hamiltonian cycle in $\mathcal{H}$.
3. In case $c = 0$, Vic verifies whether σ applied to $\mathcal{G}$ yields $\mathcal{H}$. In case $c = 1$, he checks if the uncovered entries are all ones and whether they form a Hamiltonian cycle.

The protocol is 2-extractable: It is easily seen that from the answers to both challenges (for a fixed committed graph $\mathcal{H}$), one can compute the Hamiltonian cycle. Using the idealized commitments, the protocol is also c-simulatable: For $c = 0$, choose a random permutation and (pretend to) commit to the permuted graph. The response to the challenge is simulated by (pretending to) open the entire graph and sending the permutation. For $c = 1$, commit to the all-one matrix and open a random Hamiltonian cycle.

References

[BCC88] Gilles Brassard, David Chaum, and Claude Crépeau. Minimum disclosure proofs of knowledge. *J. Comput. Syst. Sci.*, 37(2):156–189, 1988.

[1] A Hamiltonian cycle is a cycle that visits every vertex of a graph exactly once.