ETH Zurich, Department of Computer Science
SS 2018

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

Cryptographic Protocols
Solution to Exercise 10

10.1 Information-Theoretic Commitment Transfer Protocol

a) In protocol Commit the state of the dealer D consists of commit polynomial g, where the committed value is $g(0) = s$. Every player P_i stores the commit-share $s_i = g(\alpha_i)$.

b) The commitment transfer protocol CTP allows to transfer a commitment from a player P to a player P' The protocol works as follows:

1. P sends the polynomial g to P'.
2. Each P_i sends s_i to P'.
3. P' checks that all but at most t of the received s_i's lie on g. If so, he accepts $g(0)$ as value for s, otherwise he assumes that he did not receive any value for s.

The above protocol is secure for $t < n/3$:

Privacy: Straight-forward as only P' receives values in the protocol and he only obtains the values which he is supposed to receive.

Correctness: This can be argued along the lines of the correctness of the protocol Open from the lecture notes: Assume that P sends P' some wrong polynomial $g' \neq g$. Then, at most t of the commit shares can lie on polynomial g'. Hence the commit shares of at least $n - t$ players do *not* lie on g'. As at most t of those players might be corrupted, there are at least $n - 2t > t$ players who will send commit shares that do not lie on g' to P', and therefore P' will not accept $g(0)$ as value for s.

In the case that P' did not receive a valid value for s, he can accuse P via broadcast and the whole protocol is repeated, using broadcast instead of sending values.

10.2 Information-Theoretic Commitment Multiplication Protocol

In the following we will use f_a and f_b to denote the polynomials used in the commitment sharing protocol (CSP) to share the values a and b, respectively. Furthermore, let $f_c := f_a \cdot f_b$.

a) We show that correctness and privacy are satisfied:

Privacy: In steps 1-2, privacy is guaranteed by the privacy of the CSP, i.e., no information on a, b, and c is revealed in these steps. In step 3, the players only see values they already know, namely $c_i = a_i \cdot b_i$, hence again no information is revealed. Finally, the commitments to some a_i, b_i, and c_i are opened only if D or the player P_i is corrupted, which means that the adversary already knows them.

Correctness: Any dealer who is not disqualified must successfully complete the CSP for values a and b. Thus, every player P_i ends up with shares a_i on f_a and b_i on f_b. Suppose, D commits to a value $c' \neq c$ and shares it using a polynomial

$f_{c'} \neq f_c = f_a \cdot f_b$ in protocol CSP.[1] Since both f_c and $f_{c'}$ have degree at most $2t$, they can have at most $2t$ points in common. Thus, there exists at least one honest player P_i for which $c'_i \neq a_i b_i$, where c'_i is his share of c'.[2] This player will accuse the dealer and prove that he is corrupted by opening a_i, b_i, and c_i.

b) Let $n = 3t$, and assume that the players $P_1, \ldots, P_t$ are corrupted, where P_1 plays the role of D. In order to achieve that at the end of the protocol the players accept a false $c' \neq ab$, the corrupted players have the following strategy:

1. In step 0, D chooses c' (instead of c) and is commited to it.
2. Step 1 is executed normally, i.e., D invokes the CSP for a and b.
3. In step 2, D invokes the CSP for c', with the (unique) degree-$2t$ polynomial $f_{c'}(x)$, such that $f_{c'}(0) = c'$ and

$$f_{c'}(\alpha_i) = f_a(\alpha_i) \cdot f_b(\alpha_i)$$

   for $i = t+1, \ldots, n$.
4. The corrupted players do not complain in step 3.

As $f_{c'}(x)$ is chosen such that it satisfies the consistency check for all honest players, no player will complain and the commitment to c' will be accepted.

10.3 Beaver's Multiplication Triples

a) One way to create such triples is to first have each player P_i create a sharing of random values a_i and b_i, compute a sharing of $a = \sum_i a_i$ and $b = \sum_i a_i$ and finally compute a sharing of $c = ab$.

b) Assume that we are given sharings of x and y and want to compute a sharing of xy. Let a, b, c be a multiplication triple. The party computes a sharing of $x - a$ and $y - b$ and reconstruct $\alpha = x - a$ and $\beta = y - b$. Observe that since a, b are uniformly random, α and β are also random values. Moreover, observe that $xy = ab + \alpha b + a\beta + \alpha\beta$, and hence each player P_i can compute locally a sharing of xy as follows: $[xy]_i = [c]_i + \alpha[b]_i + \beta[a]_i + \alpha\beta$. Observe that only two reconstructions (and some local computation) are needed to execute this (send n^2 values sent in total), whereas in the CMP protocol seen in the lecture one needs to execute one CSP per party (each costs t broadcasts and n values sent) and in the case of cryptographic security the distributed zero-knowledge proofs, and in the case of information theoretic CMP three CSP more per party.

To modify the MPC protocol seen in the lecture, one would consider an off-line phase where the parties create a large number of multiplication triples (assuming that an upper bound on the number of multiplication gates of the function is known). Then, in the on-line phase the players do the same as in the protocol seen in the lecture, except when evaluating multiplication gates, in which the parties execute the protocol described above.

---

[1] Note that the dealer cannot share c' using f_c as can easily be seen by inspecting the CSP.

[2] The condition $t < n/3$ implies that there are at least $2t + 1$ honest players.