CS 276 – Cryptography March 8, 2016

Lecture 15: MPC in the head

Instructor: Sanjam Garg *Scribe: Akshayaram Srinivasan*

1 Introduction

Zero knowledge proofs are one of the most used fundamental building blocks in Cryptography. Recall that zero knowledge proofs allow a prover to prove to a verifier about the veracity of a statement without revealing anything beyond the assertion. Zero knowledge proofs have notably found several applications in the area of secure multiparty computation. Secure multiparty computation allows a set of n mutually distrusting parties to compute a joint function of their private inputs. The security guarantee is that any algorithm that corrupts even all but one of the n parties cannot learn anything about the input of the uncorrupted parties beyond what it can learn from the function output. The security of MPC comes in different flavors depending on "how" the parties are corrupted. If the corruption algorithm follows the protocol description but may try to learn arbitrary information from the protocol transcripts then we call it as the *semi-honest* setting. On the other hand, if the corruption algorithm can deviate arbitrarily from the protocol description then we term it as *malicious* setting. Notice that malicious MPC even for the case of two parties directly implies Zero Knowledge protocol. [1] In 2007, Ishai, Kushilevitz, Ostrovsky and Sahai proved a surprising result that even semi-honest MPC is sufficient to obtain Zero Knowledge proofs. Additionally the Zero Knowledge protocol obtained as a result of their transformation has some "nice" properties like low communication complexity.

2 Properties of Secure MPC

In this section, we describe the properties that the secure MPC must satisfy so that it can be compiled to a zero knowledge proof.
Let $P_1, \cdots, P_n$ be the n-parties. All of them share a common public input x and each party P_i has a private input w_i. The functionality that they wish to compute is as follows: let L be a NP language and R_L denote the corresponding witness relation. Parties intend to check if $R_L(x, \oplus_{i\in[n]} w_i) = 1/0$. Let us assume that the parties wish to compute the intended functionality using the protocol Π. We assume that Π is specified interms of the next message function. That is, given $(i, x, w_i, r_i, (m_1, \cdots, m_j))$, Π where r_i denotes the random coins, $m_1, \cdots, m_j$ denote the messages that the party received in the first j rounds, the next message function outputs the set of n-messages that the party P_i has to send to every other party in round $j+1$. We would define View of party P_i to constitute the private input w_i, randomness r_i and the set of messages received and sent by party P_i during the execution of the protocol.

Definition 1 *We say two views* View_i *and* View_j *of parties* P_i *and* P_j *are consistent (with respect*

[1] Zero knowledge can be viewed as special function evaluation. The two parties have a common input x whereas the prover has a private input w which is a witness to the assertion that x is some NP language L. The function they want to compute is $R_L(x, w)$ which checks if w is a valid witness or not. Since for zero knowledge we assume the verifier is allowed to arbitrarily deviate from the protocol description, we need malicious security from the underlying MPC.

to a protocol Π *and common input* x*) if the set of messages* P_i *sends to party* P_j *is exactly equal to the set of messages received by party* P_j *from* P_i *and vice versa.*

We will now prove the following lemma. The lemma states that the views of every pair of parties is *locally* consistent (with respect to some protocol Π and input x) if and only if it is *globally* consistent with respect to the same protocol Π.

Lemma 1 *Let* $\mathsf{View}_1, \cdots, \mathsf{View}_n$ *be a set of views. The views* View_i *and* View_j *for every* $i, j \in [n]$ *are consistent (with respect to a protocol* Π *and common input* x*) if and only if there exists an honest execution of the protocol* Π *on a common input* x *and private inputs* w_i *for party* P_i *where the view of the party* P_i *corresponds to* View_i*.*

Proof. The If part of the proof follows directly from the definition of protocol Π. We now prove the only-if part. We prove it using the induction on the number of rounds. The base case is when the protocol Π has only 1 round in which case the assertion follows directly. Let us assume that assertion is true for d rounds. Let w_i and r_i be the private input and randomness in View_i. In the $d+1$-th round every party P_i generates a message to every other party P_j based on the protocol Π, the input x, private input w_i, randomness r_i and the set of messages $(m_1, \cdots, m_d)$ it has received in the first d rounds (follows from local consistency constraint i.e views of parties P_i and P_j are consistent with a protocol Π). Since for the first d rounds the messages are equivalent in the local view and the global view, the $d+1$-th message in the local view is equivalent to the $d+1$-th message in the global view.
We will now define the correctness requirement from the MPC protocol.

Definition 2 *We say that the protocol* Π *perfectly computes the functionality* R_L *if for all common input* x *and private inputs* $w_1, \cdots, w_n$ *the output of each party* P_i *where* $i \in [n]$ *is equal to* $R_L(x, \oplus_{i\in[n]} w_i)$ *with probability 1 where the probability is over the random inputs of each party.*

We note that the correctness requirement can be relaxed in a natural way where we require the probability over the random inputs of the parties that each party computes the correct output is $1 - \mathsf{negl}(\kappa)$.
We now define the privacy requirement.

Definition 3 (t-privacy) *We say that* Π *realizes* R_L *with perfect* t*-privacy if there exists a simulator* Sim *such that for all* $x, w_1, \cdots, w_n$*,* $T \subset [n]$ *such that* $|T| \leq t$*,*

$$\mathsf{View}_T(x, w_1, \cdots, w_n) \equiv \mathsf{Sim}(T, x, \{w_i\}_{i\in T}, R_L(x, \oplus_{i\in[n]} w_i))$$

We again relax perfect t-privacy to statistical (or computational t-privacy).
We now define a robustness condition. Looking ahead we would require t-robustness condition to get negligible soundness error without sequential repetition.

Definition 4 (t-robustness) *We say that* Π *realizes* R_L *with* t*-robustness if for all* x *the following holds: if for every unbounded adversary corrupting a set of parties denoted by* T *where* $|T| \leq t$ *and there does not exist* $w'_1, \cdots, w'_n$ *such that* $R_L(x, \oplus_{i\in[n]} w'_i) = 1$ *then all parties in* $[n] - T$ *outputs* 0*.*

- Prover chooses random $w_1, \cdots, w_n$ subject to the condition that $\oplus_{i\in[n]} w_i = w$ where w is the witness.
- Prover runs the MPC protocol Π "in his head" (by choosing uniform random coins for each party) to generate the views of each party P_i. Let View_i denote the view of party P_i in the execution of Π.
- Prover generates commitment to each View_i separately using a statistically binding commitment scheme and sends the commitment to the verifier.
- Verifier chooses random $i, j \in [n]$ and sends it to the prover.
- The prover opens the commitment to the views View_i and View_j.
- Verifier checks if the openings are correct and if it is the case it checks if the views are consistent. If they are in consistent it outputs 0 if the protocol outputs 0. Otherwise, it outputs 1.

Figure 1: Basic Protocol

3 Basic Protocol

We now assume that we have a perfectly correct MPC that satisfies t-privacy (in fact $t = 2$ is sufficient). We describe the zero-knowledge protocol.

Theorem 2 *If the protocol Π is perfectly correct and satisfies (perfect) 2-privacy then the protocol described in Figure 1 is (perfect) zero knowledge with soundness error* $1 - \frac{1}{\binom{n}{2}}$

Proof. (Sketch) The correctness guarantee is easy to verify. We now argue soundness and zero knowledge.
Suppose x is not in the language L. Hence, for all $w_1, \cdots, w_n$, $R_L(x, \oplus_{i\in[n]} w_i) = 0$. If the prover executes the protocol honestly then by perfect correctness requirement for all choices of random coins generated by the prover, the protocol outputs 0. If the prover tries to cheat then it means that he does not execute the protocol honestly. By Lemma 1, it implies there exists i and j such that the views View_i and View_j are inconsistent. Hence, the verifier can catch a cheating prover with probability $\frac{1}{\binom{n}{2}}$.
Now we show Zero-Knowledgeness. We describe a simulator that simulates the view of a cheating verifier. The simulator chooses i and j uniformly at random. It then chooses random w_i and w_j and runs the simulator for two privacy of Π and obtains View_i and View_j. It then commits to $\mathsf{View}_k = 0$ for all $k \notin \{i, j\}$ and commits to View_i and View_j. The simulator then receives i^* and j^* from the verifier (Note that the probability that the verifier sends i^* and j^* when it is given real-world commitments to views and simulated commitments is same from the hiding property.). If $i = i^*$ and $j = j^*$ it opens the views. Otherwise, it aborts. [2]. The view of the simulator is is

[2] Note that we can make multiple clones of the verifier and run each of one of them and continue only if it outputs

- Prover chooses random $w_1, \cdots, w_n$ subject to the condition that $\oplus_{i \in [n]} w_i = w$ where w is the witness.
- Prover runs the MPC protocol Π "in his head" (by choosing uniform random coins for each party) to generate the views of each party P_i. Let View_i denote the view of party P_i in the execution of Π.
- Prover generates commitment to each View_i separately using a statistically binding commitment scheme and sends the commitment to the verifier.
- Verifier chooses t random $i_1, \cdots, i_t \in [n]$ and sends it to the prover.
- The prover opens the commitment to the views $\mathsf{View}_{i_1}, \cdots, \mathsf{View}_{i_t}$.
- Verifier checks if the openings are correct and if it is the case it checks if the views are consistent. If they are in consistent it outputs 0 if the protocol outputs 0. Otherwise, it outputs 1.

Figure 2: Extended Protocol

identical to the view of the verifier if the simulator does not abort which follows from the 2-privacy condition.

Remark 1 *Note that the soundness of the protocol holds only if the* Π *is perfectly correct. If the protocol* Π *is only statistically correct the prover can obtain some choice of random coins as advice that makes the protocol to "misbehave". In order to cope with statistical correctness we do a coin tossing. The prover commits to its share of randomness of each party. The verifier then sends its own share of randomness of each party. The prover then executes the protocol* Π *using the XOR of the two random strings as randomness to each party. In the decommitment phase, the prover opens its commitment to the random shares of party* P_i *and* P_j*. The verifier finally checks if the randomness used by party* P_i *and* P_j *is correctly generated.*

4 Extending the Protocol

We now extend the protocol to have negligible soundness error without sequential repetition. We again consider the case where the protocol Π is perfectly correct.

Theorem 3 *If the protocol* Π *is perfectly correct and satisfies (perfect) 2-privacy, and is t-robust then the protocol described in Figure 2 is (perfect) zero knowledge with negligible soundness error.*

Proof. The completeness and the zero-knowledge property follows from proof of the basic protocol. We now argue soundness.

$i = i^*$ and $j = j^*$. The expected number of clones is polynomial.

Let us assume for simplicity that $t = n/2$ where n is the number of parties which is also equal to the security parameter. We remark that if the prover executes the protocol Π honestly then by perfect correctness, the verifier will always reject. We now consider the case where the prover does not execute the protocol Π honestly. Let $V_1^*, V_2^*, \cdots, V_n^*$ be the views generated by prover's execution. We define the inconsistency graph on the vertices from $[n]$ and there is an edge between i and j if and only if the views are inconsistent with respect to protocol Π and input x. We consider two cases:

- **Case-1**: The vertex cover of the inconsistency graph has size at most t. This corresponds to the case where the prover executes Π corrupting at most t parties. In this case from t robustness criterion implies that the prover can cheat with probability $\frac{1}{\binom{n}{t}}$ which is negligible.
- **Case-2**: The vertex cover of the inconsistency graph has size greater than t. There exists a matching of size greater than $t/2$ in the inconsistency graph. If the verifier picks at least one edge of this matching then he can catch the cheating prover. We now argue that the probability that no edge of the matching is picked is negligible. Probability that no edge of the matching is chosen is at most $\sum_{r \in [t/2]} \binom{n/2}{t-r}\binom{t/2}{r} 2^r / \binom{n}{t}$ which is negligible.