CS 4501-6501 Topics in Cryptography	Feb 23 2015
Lecture 15	
Lecturer: Mohammad Mahmoody	*Scribe: Sumit Narain*

1 Introduction

The problem that is being tackled in this paper is where one party, known as the delegator, wants the computation of a function F to another party known as the adversary (ADV). The issue is that the delegator might not trust the adversary's answer, so the adversary should prove to the delegator that the computation was performed correctly. We want computing $F(x)$ to take $T(x)$ time. We have a preprocessing phase which should be ran once but able to delegate multiple times.

2 Delegation of FHE with Sampable Distribution

Suppose input $x \leftarrow D$, where distribution is samplable. Some reasonable assumptions that we could make is that we want to delegate only one computation, only want soundness $= \frac{1}{2}$, and willing to pay a large preprocessing time.

2.1 Steps

Step 0: Verifier sample $x' \leftarrow D$ and computes $F(x') = y'$
Step 1: Challenge - Delegator (also known as Verifier) sends (x, x') in random order
Step 2: Adversary solves and sends back results to Delegator and if $F(x') = y'$ then we accept that the adversary computed the correct answer else we reject the adversaries computation.

3 Completeness and Soundness

A system is complete if the delegator gives the adversary a input F with an input x, and the delegator will accept the output from the adversary when it is correct.
Soundness is if the adversary provides an incorrect y or y' since the ADV does not know x or x' it will be y' which is wrong with probability $\geq \frac{1}{2}$.

4 Delegation without Sampability

The difference with the delegation with sampability and without sampibility is the preprocessing phase, also known as Step 0. Instead of computing $F(x)$, the delegator gets $Encryption(x) = y$ then computes $z' = Evaluator(F, y)$ which is equivalent to $\frac{Encryption(F(x'))}{z'}$
Final Decision: given(z,z') then check that the given z' matches precomputed values and reject them if they do not match.

5 Error

Error currently is $\frac{1}{2}$, but there is a way to make the error probability closer to zero, which is to add more dummy statements or questions. As the adversary has to compute more dummy inputs then that means the dummy will have to compute more things correctly, so the probability or Error is $\frac{1}{polynomial(numberofinputs)}$ but this is error is not negligible because the value of that fraction is still a polynomial.

6 Improved Delegation of FHE

Besides encrypting dummy values, "k" number of times, we should also encrypt the x, which is the value that we want the adversary to compute, value "k" number of times. This makes the chance of error incredibly small, basically negligible. For instance, the probability of error constitutes that the adversary is giving the same incorrect answer for Encryption(x) and the correct answer on the same dummy encryptions. The chances of doing so while the values are still encrypted is $\approx \frac{1}{r}$, where r is the combination of k with 2k elements.

7 Conclusion

There are other factors that can play into this scheme in ways to reduce the number of pre-response time or attempting to use two input evaluations. Although this is beyond the scope of this paper, a way of reducing heavy preprocessing time would be to have a efficient delegator with interaction. In previous papers, we have seen how to apply this using has functions and the help of the delegator. So the new idea would be to apply fully homomorphic encryption over preprocessing and delegator does preprocessing (using hash function and interactions) over encrypted values. This just just one of the ways to improve the runtime of this scenario.

8 Reference

Chung, K. M.,Kalai, Y., Vadhan, S. (2010). Improved delegation of computation using fully homomorphic encryption. In Advances in Cryptology CRYPTO 2010 (pp. 483-501). Springer Berlin Heidelberg.