Cryptography-B coursework: Paper review

Manan Vaswani

January 31, 2019

1 Introduction

At the CRYPTO2012 conference, Mike Rosulek raised an interesting question regarding black-box constructions in Cryptography, which he discussed in his paper "Must you know the code of f to securely compute f?" [1]. This paper looks at the possibility of using functions without having to compute the actual code of such a function in Multi-Party Communications.

In their 1989 paper [2], Impagliazzo and Rudich posed the important question "When do black-box constructions actually exist?". Their results showed that to prove that a secret key agreement which uses a one-way permutation as a black box is as hard as proving P $\neq$ NP. Haitner, Ishai et al. published a paper [3] that studied whether it is possible to construct general secure computation protocols that use cryptographic primitives in a black-box way. In a similar manner, [4] used black-box access to one-way functions to demonstrate constructions of two-party cryptographic protocols. Interestingly, the topic of black-box function evaluation has not been as widely researched as one would expect due to the focus being on removing the non-black-box techniques from underlying cryptographic primitives in the protocol.

Black-box constructions in cryptography are those that rely only on the input and output behaviour of their components without actually knowing the details and construction of the components. They are widely used in cryptography due to the fact that they are highly practical, efficient and modular. Therefore it is important to note that the question asked in the title of the paper is not merely one of theoretical interest, but also has an important practical impact.

Secure Multi-Party Computation(MPC) allows mutually distrusting parties to compute a function f on its shared inputs. One non-black box step that is used in all secure MPC communications is the evaluation of this function f. The function is first expressed as a low-level circuit, and then evaluated gate by gate on the inputs provided. This means that the complexity of the communication protocol is directly dependent on the circuit complexity of the function. However, this step is unavoidable for most general purpose MPC, but the paper looks at exploring for which special purpose secure communication tasks it would be possible to have a true black box construction. In particular, the author defines the necessary conditions for these special MPC tasks and gives a proof to show why these use true black-box evaluations without actually computing the function in the blackbox.

2 Main Results

2.1 Functionally Black Box Protocols

For a general-purpose MPC with a fixed functionality f, the protocol directly depends on f anyway, so the protocol could simply have the circuit for f hard-coded and use that every time. Instead, the author models a protocol as a pair of oracle machines that is instantiated with any functionality f that is taken from a much larger class of functionalities $\mathcal{C}$, and then emulates a functionality related to f. If this class of functionalities is particularly large, the protocol cannot construct the circuit representations of all the related functionalities. Hence, these protocols "do not know" the code of their target function. With this goal, he introduces the definition of a functionally-black box protocol.

Let $\mathcal{C}$ be a class of functions. $\mathcal{F}$ is an ideal functionality implemented as an oracle machine. A **functionally-black-box (FBB)** protocol for $\mathcal{F}^{\mathcal{C}}$ (i.e $\mathcal{F}$ instantiated with $\mathcal{C}$) is a pair of interactive oracle machines (π_A, π_B) if for all $f \in \mathcal{C}$, the protocol (π_A^f, π_B^f) is a secure protocol for the ideal functionality $\mathcal{F}^f$. Here, the protocol $\mathcal{F}^f$ *must* treat the function it uses as a black-box, as it could really be any function out of the large class of functions $\mathcal{C}$.

In the definition, the security condition is guaranteed separately for each instantiation of the protocol with the different fs. Hence, for an FBB protocol, the adversaries may have access to an explicit representation of the function f that the protocol is instantiated with, so there is no compromise on the security condition being observed, but the honest parties only use a black-box definition of the function. The intent of the definition above is to characterize the efficiency of the honest parties without affecting any security conditions.

An observation is that the set $\mathcal{C}$ must not be learnable in the sense that the circuit representations of $f \in \mathcal{C}$ can be obtained by repeated interactions between the honest parties, or with an external oracle. Additionally, for all $f \in \mathcal{C}$, the domain size must be infinite, as for a constant-sized domain, $\mathcal{C}$ would be learnable by exhaustively querying the functions. If the class of functions $\mathcal{C}$ was learnable, the protocol could simply obtain the code for the functions and construct their circuits to carry out the task in a non-blackbox manner. This is precisely what we are trying to avoid, hence this condition of non-learnability is necessary. Informally, it is possible to "securely compute f without knowing the code of f" if f belongs to some class $\mathcal{C}$ that admits FBB secure protocols but is not learnable from oracle queries.

2.2 2-Hiding Autoreducible

Now the class of functions $\mathcal{C}$ in the definition of FBB protocols above is first spoken about rather arbitrarily. If the definition for every FBB protocol was feasible and secure for any and every $\mathcal{C}$, the paper would almost be trivial and this would mean that every single MPC protocol is truly blackbox which is rather absurd! The author describes a classification of such classes of functions, using the notion of autoreducibility and then shows that this classification is a sufficient condition to characterise the feasibility of MPC FBB protocols.

A class of functions is said to be autoreducible if there exists some oracle machine M such that for all $f \in \mathcal{C}$, $M^f(x) = f(x)$, and the oracle queries made by the machine are independent of x [5]. Rosulek introduces a variant of this called 2-hiding autoreducible. The notion of **2-hiding autoreducibility** requires that there exists an oracle machine M such that $M^f(x, y) = f(x, y)$ and additionally, half the oracle queries are independent of y (called type-1 queries), and the other half are independent of x (type-2 queries). This additional condition in the definition is necessary as it prevents the machine from simply querying the oracle on its input (x, y) and returning the result of this query as its output.

2.3 Theorem

The main theorem of the paper which gives a classification for semi-honest security says that there is an FBB protocol for $\mathcal{F}^{\mathcal{C}}_{\text{SFE}}$ that is secure against semi-honest PPT adversaries in the $\mathcal{F}_{\text{OT}}$-hybrid model if and only if $\mathcal{C}$ is 2-hiding autoreducible. Kilian [6] showed how to base arbitrary MPC on Oblivious Transfer(OT), so the theorem in the text can assume the OT-hybrid model without loss of generality. The OT-hybrid model refers to secure computation in the presence of an ideal OT oracle as secure computation [7].

Proving the forward direction (FBB $\implies$ 2-hiding autoreducible) of this theorem is fairly straightforward using a reduction. Given an FBB protocol, if Alice is given x, and Bob is given y, the protocol outputs $f(x, y)$, with Alice and Bob making independent queries to their f oracle. This protocol can be treated as an oracle machine described in the definition of 2-hiding autoreducibility. Correctness is met as the protocol returned $f(x, y)$, and the additional condition of the oracle queries being independent of x or y is also met as Alice and Bob make their queries independently.

Proving the opposite direction (2-hiding autoreducible $\implies$ FBB)requires the trusted setup (from the $\mathcal{F}_{\text{OT}}$-hybrid model) to run a cryptographic game that has access to the oracle machine from the 2-hiding autoreducible definition. The oracle machine makes type-1 and type-2 queries by asking Alice and Bob respectively to compute these queries (which they can do, since Alice and Bob have strictly black box access to the function f, that is taken from the class $\mathcal{C}$) via the trusted third-party setup. Correctness in FBB requires Alice and Bob to be able to output $f(x, y)$, which they now can do as the oracle machine from 2-hiding-autoreducibility outputs exactly that after making its queries.

2.4 Positive Example

In the paper, Rosulek illustrates a positive example of a case where a true black-box construction can be achieved. In particular, all that needs to be done is find a class of functions $\mathcal{C}$ that is 2-hiding-autoreducible, and not learnable. The class is described with the help of a signature scheme. Let Σ = (KeyGen, Sign, Ver) be an existentially unforgeable signature scheme and $(\pi_\mathrm{C}, \pi_\mathrm{S})$ denote a blind signature protocol for this scheme. A blind signature scheme is defined to be a signature scheme in which the view of the signer is independent of the message m, and only depends on the signing key sk. A blind signature protocol is said to be modular if the π_S protocol only uses the signing key via oracle access to $\mathrm{Sign}(sk, .\ )$. The the Boneh-Lynn-Shacham signature scheme [8] is an example of a blind signature protocol that is also modular. Rosulek claims that if a signature scheme $Sigma$ has a blind, modular protocol, then the class $\mathcal{C}_\Sigma = \{\mathrm{S}_{sk} | sk \in \{0,1\}^*\}$ where $\mathrm{S}_{sk}(x, y) = \mathrm{Sign}(sk, x)$, is 2-hiding autoreducible but not learnable. The proof is carried out by constructing a machine M that simulates the signature protocol, and satisfies the definition of 2-hiding autoreducibility. The fact that $\mathcal{C}_\Sigma$ is not learnable follows from the fact that the signature scheme is existentially unforgeable and is shown by demonstrating that an attack is possible against existential unforgeability if the class was learnable.

2.5 Negative Example

The paper also gives an example of a negative case: a class of functions for which it is impossible to construct an FBB protocol. This is the class of all pseudorandom functions. The pseudorandom functions take two arguments: a key (which comes from Alice, to fit with the context from the paper), and a message (from Bob). Intuitively, one can see that pseudorandom functions are not 2-hiding autoreducible due to their lack of structure. To ensure the security of pseudorandom functions, the key must be chosen at random. However, with the definition of 2-hiding autoreducible, Alice provides the key and the oracle machine can query the unkeyed pseudorandom function (type-1 query). The paper gives a formal proof to show that there is not FBB protocol for the class of pseudorandom functions, under certain conditions for the security of the PRF (which is beyond the scope of the summary of this paper review).

2.6 Results for Malicious Security

Following his analysis on black-box protocols against semi-honest adversaries, the author shows results for FBB MPC protocols that are secure against malicious adversaries. The criteria for malicious security is another variant of autoreducibility, called 1-hiding autoreducibility. This is stricter than 2-hiding autoreducibility that was defined earlier on. In simple terms, a class $\mathcal{C}$ of functions is said to be 1-hiding autoreducible if for all $f \in \mathcal{C}$, there exists a PPT oracle machine M with oracle access to f, that can determine $f(x, y)$ and its oracle queries are independent of x and y.

He uses this, and states with a proof sketch that if $\mathcal{C}$ is 1-hiding autoreducible, then there exists an FBB protocol that is secure against malicious adversaries for $\mathcal{F}_{\mathrm{SFE}}^{\mathcal{C}}$ (the functionality describing secure function evaluation). A notable difference between this statement and the parallel statement for semi-honest adversaries is that this one is a one-way implication whereas in the case for semi-honest adversaries the statement said that 2-hiding autoreducibility and the existence of an FBB protocol were equivalent. In practice, this shouldn't make a difference as the end goal we are usually trying to show is that an FBB functionality exists, given some non-black-box functionality and a class of functions.

For arbitrary functionalities against malicious security, the paper gives another theorem that uses an FBB protocol for $\mathcal{F}^{\mathcal{C}}_{\mathrm{FZK}}$ secure against malicious adversaries to yield an FBB protocol secure against malcious security for an arbitrary functionality $\mathcal{F}^{\mathcal{C}}$. The proof uses a commitment scheme to ensure consistency of each parties responses in the $\mathcal{F}^{\mathcal{C}}$ protocol and the reduction is based on a variant of the GMW compiler [9].

2.7 Zero Knowledge Proof

Finally, the author showed that there is no standalone-secure FBB, honest-verifier witness-hiding protocol for $\mathcal{F}^{\mathcal{C}_{\mathrm{OWF}}}_{\mathrm{FZK}}$ (where $\mathcal{C}_{\mathrm{OWF}}$ denotes the class of all one-way functions) under the condition that injective one-way functions exist.

3 Conclusion

In his text, Rosulek has succeeded in showing that it is possivle to "evaluate f without knowing the code of f", but only under certain conditions. Although the proof and results were interesting and seem like they have potential, it was cited only a few times in future works. One possible reason is that the cases where it can be applied are quite specific, so the theorems provided are not of much use to most cryptographers. One interesting application of the work appeared in [10], in which a blockchain protocol uses a 'channel' functionality in an FBB manner. In [11], the authors proved an unconditional version the impossibility result of Rosuleks theorem on FBB protocols for Zero Knowledge proofs (Section 2.7). Another recent paper [12] investigates the question of whether there exists a FBB round-preserving malicious protocol for the parallel composition of probabilistic-termination functionalities, which illustrates the usefulness of Rosuleks definition of FBB protocols.

References

[1] Mike Rosulek. Must you know the code of f to securely compute f? In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 87–104. Springer, Heidelberg, August 2012.

[2] Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In *21st ACM STOC*, pages 44–61. ACM Press, May 1989.

[3] Iftach Haitner, Yuval Ishai, Eyal Kushilevitz, Yehuda Lindell, and Erez Petrank. Black-box constructions of protocols for secure computation. Cryptology ePrint Archive, Report 2010/164, 2010. `http://eprint.iacr.org/2010/164`.

[4] Rafael Pass and Hoeteck Wee. Black-box constructions of two-party protocols from one-way functions. In Omer Reingold, editor, *TCC 2009*, volume 5444 of *LNCS*, pages 403–418. Springer, Heidelberg, March 2009.

[5] Harry Buhrman, Lance Fortnow, and Leen Torenvliet. Using autoreducibility to separate complexity classes. In *36th FOCS*, pages 520–527. IEEE Computer Society Press, October 1995.

[6] Joe Kilian. Founding cryptography on oblivious transfer. In *20th ACM STOC*, pages 20–31. ACM Press, May 1988.

[7] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer - efficiently. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 572–591. Springer, Heidelberg, August 2008.

[8] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. *Journal of Cryptology*, 17(4):297–319, September 2004.

[9] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th ACM STOC*, pages 218–229. ACM Press, May 1987.

[10] Christian Badertscher, Ueli Maurer, Daniel Tschudi, and Vassilis Zikas. Bitcoin as a transaction ledger: A composable treatment. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 324–356. Springer, Heidelberg, August 2017.

[11] Yuval Ishai, Eyal Kushilevitz, Manoj Prabhakaran, Amit Sahai, and Ching-Hua Yu. Secure protocol transformations. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part II*, volume 9815 of *LNCS*, pages 430–458. Springer, Heidelberg, August 2016.

[12] Ran Cohen, Sandro Coretti, Juan A. Garay, and Vassilis Zikas. Round-preserving parallel composition of probabilistic-termination cryptographic protocols. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *ICALP 2017*, volume 80 of *LIPIcs*, pages 37:1–37:15. Schloss Dagstuhl, July 2017.