Brief Announcement: Impossibility Results for Concurrent Secure Two-Party Computation

Yehuda Lindell*

This brief announcement describes two impossibility results regarding the concurrent composition of secure two-party protocols in the *plain model* (i.e., where no setup assumption like a common reference string is assumed).

In the setting of two-party computation, two parties wish to jointly compute some function of their private inputs. Loosely speaking, the security requirements are that nothing is learned from the protocol other than the output (privacy), and that the output is distributed according to the prescribed functionality (correctness). These requirements must also hold in the face of malicious adversarial behavior. Powerful feasibility results have been shown for this problem, demonstrating that *any* two-party probabilistic polynomial-time functionality can be securely computed [5, 2]. However, these feasibility results relate only to the stand-alone setting, where a single pair of parties run a single execution. A more general (and realistic) setting relates to the case that many protocol executions are run simultaneously (or *composed*) in a network. Unfortunately, the security of a protocol in the stand-alone setting does not imply its security under composition. Therefore, an important research goal is to reexamine the feasibility of secure computation in this setting. In this announcement, we consider the feasibility of obtaining security under *concurrent* composition where the scheduling of the protocol executions, including when they start and the rate at which they proceed, is maliciously determined by the adversary.

We distinguish between two types of composition, where the distinction relates to the question of *which protocols* are being run together in the network, or in other words, with which protocols should the protocol in question compose. The first type, called self composition considers the case where a single protocol is executed many times in a network. We stress that in this setting, there is only one protocol that is being run. The second type, called general composition considers the case where many different protocols are run together in the network. Furthermore, these protocols may have been designed independently of one another. There is a crucial difference between self and general composition. In self composition, the protocol designer has control over everything that is being run in the network. However, in general composition, the protocol designer has no control whatsoever over the other protocols being run (indeed, these protocols may even have been designed maliciously after the secure protocol is fixed). The summary of our results is as follows:

Impossibility of general composition. A protocol that remains secure under concurrent general composition can be plugged into any scenario, and its security is guaranteed. Protocols that remain secure under this type of composition would be very desirable. We show that there are large classes of functionalities for which security under concurrent (or even parallel) general composition *cannot* be achieved [4]. We stress that this impossibility result is unconditional; it does not rely on complexity assumptions and is not a "black-box" lower bound. Note that this result holds only for the plain model, where no trusted setup phase is assumed. (In contrast, in the common reference string model, there do exist protocols that remain secure under concurrent general composition [1].)

Black-box lower bounds for self composition. A far weaker type of composition is that of self composition. Nevertheless, we show that under the assumption that one-way functions exist, there exist (natural) functionalities for which the following holds: Every secure protocol for computing these functionalities that remains secure when self-composed m times and is proven using black-box simulation, must have more than m rounds of communication [3]. Thus, protocols that remain (black-box) secure for *any* concurrent number of executions *do not exist* for these functionalities. This is in stark contrast to the situation for concurrent zero-knowledge (which is also stated in the context of self composition). Unlike the impossibility result for general composition, this lower bound leaves open the possibility of obtaining positive results. However, it does show that even in this rather weak setting of concurrency, obtaining security is "difficult".

We conclude with a caveat. Our results are proven for specific, natural definitions of security under composition. These definitions seem to be as weak as possible, while still capturing the notion of security under composition. Nevertheless, it is possible that different definitions can be formulated for which the impossibility results will not hold. In particular, we do not consider definitions which are not formulated via the simulation paradigm.

*IBM T.J.Watson Research, email: `lindell@us.ibm.com`

In the 22nd ACM Symposium on the Principles of Distributed Computing (PODC), 2003.

REFERENCES

[1] R. Canetti, Y. Lindell, R. Ostrovsky and A. Sahai. Universally Composable Two-Party and Multi-Party Computation. In 34*th STOC*, pages 494–503, 2002.

[2] O. Goldreich, S. Micali and A. Wigderson. How to Play any Mental Game – A Completeness Theorem for Protocols with Honest Majority. In *19th STOC,* pages 218–229, 1987.

[3] Y. Lindell. Bounded-Concurrent Secure Two-Party Computation Without Setup Assumptions. In 35*th STOC*, 2003.

[4] Y. Lindell. General Composition and Universal Composability in Secure Multi-Party Computation. To appear in 44*th FOCS*, 2003.

[5] A. Yao. How to Generate and Exchange Secrets. In 27*th FOCS*, pages 162–167, 1986.