# Preface to Special Issue on General Secure Multi-Party Computation

Oded Goldreich
Department of Computer Science
Weizmann Institute of Science
Rehovot, ISRAEL.
oded@wisdom.weizmann.ac.il

October 26, 1999

More than a decade has passed since general results concerning secure two-party and multi-party computations were first announced in [15, 24, 16] (see details in [14]). In a nutshell, assuming the existence of trapdoor permutations, these results assert that one can construct protocols for securely computing any desirable multi-party functionality. These results either require a majority of honest players or allow dishonest players to suspend the execution (while being detected as bad). Subsequent "unconditional" results in the "private channel model" require a 2/3-majority of honest player (cf., [4, 7]).

The abovementioned results were presented at a time in which *intensive* electronic multi-party interactions seemed a remote possibility. So it seems fair to say that, while generating considerable interest within the theory community, these results generated little interest in the applied cryptography community. But times have changed: Intensive electronic multi-party interactions seems almost a reality, and the entire cryptographic community seems very much interested in a variety of natural problems which arise from such a reality. This has triggered the idea of having a special issue on general secure multi-party computation.

Most of the current application-oriented interest is focused on the design of efficient and secure schemes for *specific* purposes – Threshold Cryptography (cf., [10, 12]) is indeed a good example. Nevertheless, we believe that the *general* study of secure multi-party computation is important to practice. Firstly, this study clarifies fundamental issues regarding security in a multi-party environment. Secondly, it draws the lines between what is possible in principle and what is not. Thirdly, it develops general techniques for design of secure protocols. And last, sometimes, it may even yield schemes (or modules) which may be incorporated in practical systems. Thus, we believe that the current special issue may be both of theoretical and practical interest.

In order to put the papers of this special issue in perspective, it seems good to start with a short overview of the area.

## Background on General Secure Multi-Party Computation

A general framework for casting cryptographic (protocol) problems consists of specifying a random process which maps $m$ inputs to $m$ outputs. The inputs to the process are to be thought of as local inputs of $m$ parties, and the $m$ outputs are their corresponding local outputs. The random process describes the desired functionality. That is, if the $m$ parties were to trust each other (or trust some outside party), then they could each send their local input to the trusted party, who would compute the outcome of the process and send to each party the corresponding output. A pivotal question in the area is to what extent can this (imaginary) trusted party be "emulated" by the mutually distrustful parties themselves.

The results mentioned in the first paragraph above (as well as many subsequent results) describe a variety of models in which such emulation is possible. The models vary by the underlying assumptions regarding the communication channels, numerous parameters relating to the extent of adversarial behavior, and the desired level of emulation of the trusted party (i.e., level of "security").

Before describing these results, a few words regarding the notion of "emulating a trusted party" are in place. This notion is the pivot of the basic approach to defining secure multi-party computation as initiated in [17, 20, 1, 2].[1] The approach can be traced back to the definition of zero-knowledge (cf., [19]), and even to the definition of secure encryption (cf., [13] rephrasing [18]). The underlying paradigm is that a scheme is secure if whatever a feasible adversary can obtain after attacking it, is also feasibly attainable from scratch. In case of zero-knowledge this amounts to saying that whatever a (feasible) verifier can obtain after interacting with the prover on a prescribed valid assertion, can be (feasibly) computed from the assertion itself. In case of multi-party computation we compare the effect of adversaries which participate in the execution of the actual protocol to the effect of adversaries which participate in an imaginary execution of a trivial (ideal) protocol for computing the desired functionality with the help of a trusted party. If whatever adversaries can feasibly obtain in the former actual setting can also be feasibly obtained in the latter ideal setting then the protocol "emulates a trusted party" (and so is deemed secure). This basic approach can be applied in a variety of models, and is used to define the goals of security in these models. (For more details see Canetti's paper in this special issue.) We next discuss some of the parameters used in defining these models.

- *The communication channels*: A useful abstraction is one of a "private channel" (cf., [4, 7]) as opposed to a channel which the adversary may be tapping. That is, it is postulated that the adversary cannot obtain messages sent between a pair of honest players. In addition, one may postulate the existence of a broadcast channel (cf., [22]). Clearly, both types of channels can be emulated by ordinary "tapped channels" under standard assumptions, but the point is that the former provide a clean model for study and development of secure protocols.

- *Computational limitations*: Typically, we consider computationally-bounded adversaries (e.g., probabilistic polynomial-time adversaries), however the private-channel model allows us also to (meaningfully) consider computationally unbounded adversaries.

  We stress that security in the latter case should be defined by saying that for every adversary, whatever it can compute after participating in the execution of the actual protocol is computable *within comparable time* by an imaginary adversary participating in an imaginary execution of the trivial protocol (for computing the desired functionality with the help of a trusted party). Thus, results in the computationally unbounded adversary model trivially imply results for computationally-bounded adversaries.

- *Restricted adversarial behavior*: The most general type of an adversary considered in the literature is one which may corrupt parties to the protocol while the execution goes on, and

[1] Our current understanding is most indebted to the high-level discussions in Micali and Rogaway's unfinished manuscript (cf., [20]). Beaver's papers [1, 2] have a similar approach. The approach of Goldwasser and Levin [17] is more general: It avoids the definition of security (w.r.t a given functionality) and instead defines a notion of protocol robustness.

do so based on partial information it has gathered so far (cf., [5]). A somewhat more restricted model, which seems adequate in many setting, postulates that the set of dishonest parties is fixed (arbitrarily) before the execution starts. The latter model is called *non-adaptive* as opposed to the *adaptive* adversary discussed first.

An orthogonal parameter of restriction refers to whether a dishonest party takes active steps to disrupt the execution of the protocol (i.e., sends messages other than instructed), or merely gathers information which it may latter share with the other dishonest players. The latter adversary has been given a variety of names such as *semi-honest*, *passive*, and *honest-but-curious*. (Such an adversary is analogous to the honest-verifier considered in the zero-knowledge literature.) This restricted model may be justified in certain settings, and certainly provides a useful methodological locus (cf., [15, 16, 14]).

- *Restricted notions of security*: One example is the willingness to tolerate "unfair" protocols in which the execution can be suspended by a dishonest player, provided that it is detected doing so. We stress that in case the execution is suspended, the dishonest party does not obtain more information than it could have obtained when not suspending the execution.
- *Upper bounds on the number of dishonest parties*: In some models secure multi-party computation is possible only if a majority of the players are honest (cf., [4, 9]). Sometimes even a special majority (e.g., 2/3) is required.
- *Mobile adversary*: In most works, once a party is said to be dishonest it remains so throughout the execution. More generally, one may consider transient adversarial behavior (e.g., an adversary seizes control of some site and later withdraws from it). This model, introduced in [21], allows to consider protocols which remain secure even in case the adversary may seize control of all sites during the execution (but never control concurrently, say, more than 33% of the sites). We comment that schemes secure in this model were later termed "proactive" (cf., [6]).

We next mention some of the models for which general secure multi-party computation is known to be attainable.

- *Assuming the existence of trapdoor permutations*, secure multi-party computation is possible in the following models (cf., [16] and [14]):

   1. Passive adversary, for any number of dishonest parties.
   2. Active adversary which may control only a minority of the parties.
   3. Active adversary, for any number of bad parties, provided that suspension of execution is not considered a violation of security (i.e., see restricted notion of security above).

   In all cases, the adversary is computationally-bounded and non-adaptive. It may tap the communication lines between honest parties (i.e., we do not assume "private channels").

- Making no computational assumptions and allowing computationally *un*bounded adversaries, but *assuming private channels*, secure multi-party computation is possible in the following models (cf., [4, 7]):

   1. Passive adversary which may control only a minority of the parties.

2. Active adversary which may control only less than one third of the parties.[2]

In both cases the adversary may be adaptive (cf., [4, 5]).

- Secure multi-party computation is possible against an active, adaptive and *mobile* adversary which may control a small constant fraction of the parties at any point in time [21]. This result makes no computational assumptions, allows computationally *un*bounded adversaries, but *assumes private channels.*
- *Assuming the intractability of inverting RSA or of the DLP*, secure multi-party computation is possible in a model allowing an *adaptive* and active computationally-bounded adversary which may control only less than one third of the parties [5]. We stress that this result does not assume "private channels".

## The actual papers in this special issue

In 1997, the *ACM Annual Symposium on Principles of Distributed Computing* (PODC) and the *Annual International Cryptology Conference* (CRYPTO) have co-located at Santa Barbara, which is CRYPTO's regular location. This fact reflects the deep connections between the two areas. The idea of having a special issue on secure multi-party computation has occured to us at that occasion. In fact, three of the five papers in this special issue were presented at the PODC97 conference.

**Secure Communication in Minimal Connectivity Models (by Franklin and Wright):** Following [11], this work focuses on the communication model and studies under what conditions can one implement "private channels" over an arbitrary network. That is, not every pair of parties is linked via a direct communication line, and communication between such "unlinked" pairs has to be routed via other parties. The work considers traditional communication lines as well as "multi-cast lines" and broadcast channels. It also contrasts perfect security with almost perfect security.

**Player Simulation and General Adversary Structures in Perfect Multi-Party Computation (by Hirt and Maurer):** This work views previous work on multi-party computation as focused on a very specific "adversary structure" (family of possible coalitions of dishonest players); that is, threshold structures. This uniform view of parties is not realistic, and it has a cost; in order to protect against some likely coalition of dishonest parties one needs to protect against all coalitions of that size. Instead, the current work deals with arbitrary adversary structures which represent the set of possible coalitions one wants to protect against. It is shown that one can protect against adversary structures that are only "covered" by threshold structures which do not allow general multi-party computation: Consider, for example, general $m$-party computation in presence of an active, computationally unbounded adversary (with private-channels as in [4, 7]). Then this work shows how to provide $m$-party computation which is secure both against a possible coalition of $m/2$ specific parties as well as against all coalitions of less than $m/4$ parties. (Recall that one cannot possibly protect against all possible coalitions of $m/3$ parties.)

[2] Fault-tolerance can be increased to a regular minority if broadcast channels exists [22].

**Maintaining Authenticated Communication in the Presence of Break-Ins (by Canetti, Halevi and Herzberg):** Typical work regarding general secure multi-party computation typically assumes that the "parties know who they are talking to" (i.e., the channels between them are authenticated). In the traditional setting, this assumption can be easily justified by using public-key cryptosystems. The current work deals with a setting (as in [21, 6]) in which an adversary may temporarily gain control of all network sites, provided it never controls too many sites concurrently. The work shows how to maintain authenticated communication in such an setting.

**Randomness versus Fault-Tolerance (by Canetti, Kushilevitz, Ostrovsky, and Rosen):** This work provides an upper bound on the amount of (perfect) randomness required for general secure $m$-party computation. Letting $t$ denote an upper bound on the number of dishonest parties, the saving obtained is typically a reduction of a multiplicative factor of poly$(m/t)$. This is obtained by partitioning the computation among disjoint teams of size $O(t)$, and "recycling" randomness among the teams. Thus, it is advantageous to partitioning work among all parties, rather than let a single designated team of $O(t)$ parties do all the work. The presentation benefits from the composition theorems presented in the next paper.

**Security and Composition of Multi-party Cryptographic Protocols (by Canetti):** This work provides a relatively simple, flexible and comprehensive treatment of the definitions of secure multi-party computation. The current work may be viewed as a minimalistic instantiation of the definitional approach of [20], where minimality refers to possible augmentations of the high-level approach as presented above. In the past, it was believed that several restricting augmentations (such as "one-pass black-box simulability") are necessary in order to ensure modularity (i.e., composition theorems regarding secure protocols). The current work show that the minimalistic approach suffices for sequential composition theorems.

## References

[1] D. Beaver. Foundations of Secure Interactive Computing. In *Crypto91*, Springer-Verlag LNCS (Vol. 576), pages 377–391.

[2] D. Beaver. Secure Multi-Party Protocols and Zero-Knowledge Proof Systems Tolerating a Faulty Minority. *J. Cryptology*, Vol. 4, pages 75–122, 1991.

[3] D. Beaver, S. Micali and P. Rogaway. The Round Complexity of Secure Protocols. In *22nd STOC*, pages 503–513, 1990.

[4] M. Ben-Or, S. Goldwasser and A. Wigderson. Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation. In *20th STOC*, pages 1–10, 1988.

[5] R. Canetti, U. Feige, O. Goldreich and M. Naor. Adaptively Secure Multi-party Computation. In *28th STOC*, pages 639–648, 1996.

[6] R. Canetti and A. Herzberg. Maintaining Security in the Presence of Transient Faults. In *Crypto94*, Springer-Verlag LNCS (Vol. 839), pages 425–439.

[7] D. Chaum, C. Crépeau and I. Damgård. Multi-party unconditionally Secure Protocols. In *20th STOC*, pages 11–19, 1988.

[8] B. Chor, S. Goldwasser, S. Micali and B. Awerbuch. Verifiable Secret Sharing and Achieving Simultaneity in the Presence of Faults. In *26th FOCS*, pages 383–395, 1985.

[9] B. Chor and E. Kushilevitz. A Zero-One Law for Boolean Privacy. *SIAM J. on Disc. Math.*, Vol. 4, pages 36–47, 1991.

[10] Y. Desmedt and Y. Frankel. Threshold Cryptosystems. In *Crypto89*, Springer-Verlag LNCS (Vol. 435), pages 307–315.

[11] D. Dolev, C. Dwork, O. Waarts, and M. Yung. Perfectly secure message transmission. *J. of the ACM*, Vol. 40 (1), pages 17–47, 1993.

[12] P.S. Gemmell. An Introduction to Threshold Cryptography. In *CryptoBytes*, RSA Lab., Vol. 2, No. 3, 1997.

[13] O. Goldreich. A Uniform Complexity Treatment of Encryption and Zero-Knowledge. *Journal of Cryptology*, Vol. 6, No. 1, pages 21–53, 1993.

[14] O. Goldreich. *Secure Multi-Party Computation.* In preparation, 1998. Working draft available from `http://theory.lcs.mit.edu/~oded/gmw.html`.

[15] O. Goldreich, S. Micali and A. Wigderson. Proofs that Yield Nothing but their Validity or All Languages in NP Have Zero-Knowledge Proof Systems. *J. of the ACM*, Vol. 38, No. 1, pages 691–729, 1991. Preliminary version in *27th FOCS*, 1986.

[16] 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.

[17] S. Goldwasser and L.A. Levin. Fair Computation of General Functions in Presence of Immoral Majority. In *Crypto90*, Springer-Verlag LNCS (Vol. 537), pages 77–93.

[18] S. Goldwasser and S. Micali. Probabilistic Encryption. *J. of Comp. and Sys. Sci.*, Vol. 28, No. 2, pages 270–299, 1984. Preliminary version in *14th STOC*, 1982.

[19] S. Goldwasser, S. Micali and C. Rackoff. The Knowledge Complexity of Interactive Proof Systems. *SIAM J. on Comput.*, Vol. 18, pages 186–208, 1989. Preliminary version in *17th STOC*, 1985.

[20] S. Micali and P. Rogaway. Secure Computation. In *Crypto91*, Springer-Verlag LNCS (Vol. 576), pages 392–404. Ellaborated working draft available from the authors.

[21] R. Ostrovsky and M. Yung. How to Withstand Mobile Virus Attacks. In *10th PODC*, pages 51–59, 1991.

[22] T. Rabin and M. Ben-Or. Verifiable Secret Sharing and Multi-party Protocols with Honest Majority. In *21st STOC*, pages 73–85, 1989.

[23] A.C. Yao. Protocols for secure computations (extended abstract). In *23rd FOCS*, pages 160–164, 1982.

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