Brief Announcement: Optimally-Resilient Unconditionally-Secure Asynchronous Multi-Party Computation Revisited

Ashish Choudhury
International Institute of Information Technology Bangalore, India
ashish.choudhury@iiitb.ac.in

Abstract

In this paper, we present an *optimally-resilient*, unconditionally-secure *asynchronous multi-party computation* (AMPC) protocol for n parties, tolerating a *computationally unbounded* adversary, capable of corrupting up to $t < \frac{n}{3}$ parties. Our protocol needs a communication of $\mathcal{O}(n^4)$ field elements per multiplication gate. This is to be compared with previous best AMPC protocol (Patra et al, ICITS 2009) in the same setting, which needs a communication of $\mathcal{O}(n^5)$ field elements per multiplication gate. To design our protocol, we present a simple and highly efficient *asynchronous verifiable secret-sharing* (AVSS) protocol, which is of independent interest.

2012 ACM Subject Classification Security and privacy → Information-theoretic techniques; Theory of computation → Distributed algorithms; Theory of computation → Cryptographic protocols

Keywords and phrases Verifiable Secret-sharing, Secure MPC, Fault-tolerance, Byzantine faults, secret-sharing, unconditional-security, privacy

Digital Object Identifier 10.4230/LIPIcs.DISC.2020.44

Related Version A full version of the paper is available at [4], `https://eprint.iacr.org/2020/906`.

Funding *Ashish Choudhury*: This research is an outcome of the R & D work undertaken in the project under the Visvesvaraya PhD Scheme of Ministry of Electronics & Information Technology, Government of India, being implemented by Digital India Corporation (formerly Media Lab Asia).

1 Introduction

Secure *multi-party computation* (MPC) [9, 6, 2] is a fundamental problem in secure distributed computing. Informally a MPC protocol allows a set of n mutually-distrusting parties to perform a joint computation on their inputs, while keeping their inputs as private as possible, even in the presence of an adversary Adv who can corrupt any t out of these n parties. While the MPC problem has been pre-dominantly studied in the *synchronous* communication model where the message delays are upper bounded by a public constant, the progress in the design of efficient asynchronous MPC (AMPC) protocols is rather slow. In the latter setting, the communication channels may have arbitrary but finite delays and deliver messages in any arbitrary order, with the only guarantee that all sent messages are *eventually* delivered.

In this work, we consider a setting where Adv is *computationally unbounded*. In this setting, we have two class of AMPC protocols. *Perfectly-secure* AMPC protocols give the security guarantees without any error, while *unconditionally-secure* AMPC protocols give the security guarantees with probability at least $1 - \epsilon_{\mathsf{AMPC}}$, where ϵ_{AMPC} is any given (non-zero) error parameter. While there are quite a few works which consider optimally-resilient perfectly-secure AMPC protocol, not too much attention has been paid to the design of efficient unconditionally-secure AMPC protocol with the optimal resilience of $t < \frac{n}{3}$ [3]. In this work, we make inroads in this direction, by presenting a simple and efficient unconditionally-secure AMPC protocol.

1.1 Our Results and Comparison with the Existing Works

In any unconditionally-secure AMPC protocol (including ours), the function to be computed is abstracted as a publicly-known ckt over some finite field $\mathbb{F}$, consisting of addition and multiplication gates over $\mathbb{F}$ and the goal is to let the parties jointly and"securely" evaluate ckt. The field $\mathbb{F}$ is typically the Galois field $\mathrm{GF}(2^\kappa)$, where κ depends upon ϵ_{AMPC}. The *communication complexity* of any AMPC protocol is dominated by the communication needed to evaluate the multiplication gates in ckt. Consequently, the focus of any generic AMPC protocol is to improve the communication required for evaluating the multiplication gates in ckt. The following table summarizes the communication complexity of the existing AMPC protocols with the optimal resilience of $t < \frac{n}{3}$ and our protocol.

Reference	Communication Complexity (in bits) for a Single Multiplication Gate
[3]	$\mathcal{O}(n^{11}\kappa^4)$
[7]	$\mathcal{O}(n^5\kappa)$
This paper	$\mathcal{O}(n^4\kappa)$

We follow the standard approach of shared circuit-evaluation, where each value during the evaluation of ckt is Shamir secret-shared [8] among the parties, with threshold t. Informally, a value s is said to be Shamir-shared with threshold t, if there exists some degree-t polynomial with s as its constant term and every party P_i holds a distinct evaluation of this polynomial as its share. In the AMPC protocol, each party P_i *verifiably* secret-shares its input for ckt. The verifiability here ensures that if the parties terminate this step, then some value is indeed Shamir secret-shared among the parties on the behalf of P_i. To verifiably secret-share its input, each party executes an instance of *asynchronous verifiable secret-sharing* (AVSS). Once the inputs of the parties are secret-shared, the parties then evaluate each gate in ckt, maintaining the following invariant: if the gate inputs are secret-shared, then the parties try to obtain a secret-sharing of the gate output. Due to the linearity of Shamir secret-sharing, maintaining the invariant for addition gates do not need any interaction among the parties. However, for maintaining the invariant for multiplication gates, the parties need to interact with each other and hence the onus is rightfully shifted to minimize this cost. For evaluating the multiplication gates, the parties actually deploy the standard Beaver's circuit-randomization technique [1]. The technique reduces the cost of evaluating a multiplication gate to that of publicly reconstructing two secret-shared values, provided the parties have access to a Shamir-shared random multiplication triple (a, b, c), where $c = a \cdot b$. The shared multiplication triples are generated in advance in a bulk in a circuit-independent pre-processing phase, using the efficient framework proposed in [5]. The framework allows to efficiently and verifiably generate Shamir-shared random multiplication triples, using any given AVSS protocol. Once all the gates in ckt are evaluated and the circuit-output is available in a secret-shared fashion, the parties publicly reconstruct this value. The privacy of the computation follows from the fact that during the shared circuit-evaluation, for each value in ckt, Adv learns at most t shares, which are independent of the actual shared value. While the AMPC protocols of [3, 7] also follow the above blue-print of shared circuit-evaluation, the difference is in the underlying AVSS protocol.

AVSS is a well-known and important primitive in secure distributed computing. On a very high level, an AVSS protocol enhances the security of Shamir secret-sharing against a *malicious* adversary (Shamir secret-sharing achieves its properties only in the *passive* adversarial model, where even the corrupt parties honestly follow protocol instructions). The existing unconditionally-secure AVSS protocols with $t < n/3$ [3, 7] need high communication. This is because there are significant number of obstacles in designing unconditionally-secure

AVSS with exactly $n = 3t + 1$ parties (which is the least value of n with $t < n/3$). The main challenge is to ensure that *all honest* parties obtain their shares of the secret. We call an AVSS protocol guaranteeing this "completeness" property as *complete* AVSS. However, in the asynchronous model, it is impossible to directly get the confirmation of the receipt of the share from each party, as corrupt parties may never respond. To get rid off this difficulty, [3] introduces a "weaker" form of AVSS which guarantees that the underlying secret is verifiably shared only among a set of $n - t$ parties and up to t parties may not have their shares. To distinguish this type of AVSS from complete AVSS, the latter category of AVSS is termed an *asynchronous complete secret-sharing* (ACSS) in [3], while the weaker version of AVSS is referred as just AVSS[1]. Given any AVSS protocol, [3] shows how to design an ACSS protocol using n instances of AVSS. An AVSS protocol with $t < n/3$ is also presented in [3]. With a communication complexity of $\Omega(n^9\kappa)$ bits, the protocol is highly expensive. This AVSS protocol when used in their ACSS protocol incurs a communication complexity of $\Omega(n^{10}\kappa)$bits. Apart from being communication expensive, the AVSS of [3] involves a lot of asynchronous primitives such as ICP, A-RS, AWSS and Two & Sum AWSS. In [7], a simplified AVSS protocol with communication complexity $\mathcal{O}(n^3\kappa)$ bits is presented, based on only few primitives, namely ICP and AWSS. This AVSS is then converted into an ACSS in the same way as [3], making the communication complexity of their ACSS $\mathcal{O}(n^4\kappa)$ bits.

In this work, we further improve upon the communication complexity of the ACSS of [7]. We first design a new AVSS protocol with a communication complexity $\mathcal{O}(n^2\kappa)$ bits. Then using the approach of [3], we obtain an ACSS protocol with communication complexity $\mathcal{O}(n^3\kappa)$ bits. Our AVSS protocol is conceptually simpler and is based on just the ICP primitive and hence easy to understand. Moreover, since we avoid the usage of AWSS in our AVSS, we get a saving of $\Theta(n)$ in the communication complexity, compared to [7] (the AVSS of [7] invokes n instances of AWSS, which is not required in our AVSS). We refer the readers to [4] for the complete details of our AVSS scheme and AMPC protocol.

References

1 D. Beaver. Efficient Multiparty Protocols Using Circuit Randomization. In *CRYPTO*, volume 576 of *LNCS*, pages 420–432. Springer, 1991.

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

3 M. Ben-Or, B. Kelmer, and T. Rabin. Asynchronous Secure Computations with Optimal Resilience. In *PODC*, pages 183–192. ACM, 1994.

4 A. Choudhury. Optimally-resilient Unconditionally-secure Asynchronous Multi-party Computation Revisited. Cryptology ePrint Archive, Report 2020/906, 2020.

5 A. Choudhury and A. Patra. An Efficient Framework for Unconditionally Secure Multiparty Computation. *IEEE Trans. Information Theory*, 63(1):428–468, 2017.

6 O. Goldreich, S. Micali, and A. Wigderson. How to Play any Mental Game or A Completeness Theorem for Protocols with Honest Majority. In *STOC*, pages 218–229. ACM, 1987.

7 A. Patra, A. Choudhary, and C. Pandu Rangan. Efficient Statistical Asynchronous Verifiable Secret Sharing with Optimal Resilience. In *ICITS*, volume 5973 of *LNCS*, pages 74–92. Springer, 2009.

8 A. Shamir. How to Share a Secret. *Commun. ACM*, 22(11):612–613, 1979.

9 A. C. Yao. Protocols for Secure Computations. In *FOCS*, pages 160–164. IEEE, 1982.

[1] We stress that the weaker form of AVSS is not sufficient for the shared circuit-evaluation. This is because the set of $n - t$ share-holders might be different for different shared values.