Secure Multi-Party Computation

Fattaneh Bayatbabolghani
School of Information
University of California, Berkeley
fbayatba@ischool.berkeley.edu

Marina Blanton
Department of Computer Science and Engineering
University at Buffalo, The State University of New York
mblanton@buffalo.edu

ABSTRACT

Secure multi-party computation (SMC) is an emerging topic which has been drawing growing attention during recent decades. There are many examples which show importance of SMC constructions in practice, such as privacy-preserving decision making and machine learning, auctions, private set intersection, and others. In this tutorial, we provide a comprehensive coverage of SMC techniques, starting from precise definitions and fundamental techniques. Consequently, a significant portion of the tutorial focuses on recent advances in general SMC constructions. We cover garbled circuit evaluation (GCE) and linear secret sharing (LSS) which are commonly used for secure two-party and multi-party computation, respectively. The coverage includes both standard adversarial models: semi-honest and malicious.

For GCE, we start with the original Yao's garbled circuits construction [30] for semi-honest adversaries and consequently cover its recent optimizations such as the "free XOR," the garbled row reduction, the half-gates optimization, and the use of AES NI techniques. We follow with a discussion of techniques for making GCE resilient to malicious behavior, which includes the cut-and-choose approach and additional techniques to deter known attacks in the presence of malicious participants. In addition, we include the-state-of-the-art protocols for oblivious transfer (OT) and OT extension in the presence of semi-honest and malicious users.

For LSS, we start from standard solutions for the semi-honest adversarial model including [5, 28] and consequently move to recent efficient constructions for semi-honest and malicious adversarial models. The coverage includes different types of corruption thresholds (with and without honest majority), which imply different guarantees with respect to abort.

KEYWORDS

Secure multi-party computation, secure two-party computation, garbled circuit evaluation, secret sharing.

ACM Reference Format:
Fattaneh Bayatbabolghani and Marina Blanton. 2018. Secure Multi-Party Computation. In *2018 ACM SIGSAC Conference on Computer and Communications Security (CCS '18), October 15–19, 2018, Toronto, ON, Canada.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3243734.3264419

CCS '18, October 15–19, 2018, Toronto, ON, Canada

ACM ISBN 978-1-4503-5693-0/18/10.
https://doi.org/10.1145/3243734.3264419

1 SECURE MULTI-PARTY COMPUTATION

Secure multi-party computation (SMC) is a mature research area of computer science that has been active for decades. It also goes under the names of privacy-preserving computation and secure function evaluation (SFE). There are many common applications that benefit from this concept and demonstrate importance of SMC protocols in practice. Examples include privacy-preserving decision making on distributed medical or financial data, privacy-preserving machine learning, auctions, online poker, private set intersection of sets belonging to different organizations, and many others.

SMC allows a number of participants to securely evaluate a function on their private inputs in such a way that no information other than an agreed upon output is available to the participants. Existing solutions can be divided into two main categories based on the number of participants they support: 1) secure two-party computation and 2) secure multi-party computation. In the former setting, each of the two parties provides private input, both jointly evaluate the function, and one or both parties learn the output. In the latter setting, the secure computation is carried out by a number of computational parties. The input can come from any number of participants, the output can be delivered to a desired set of participants, and the setting supports the ability to outsource the computation by one or multiple input owners to external parties who securely carry out the computations without learning private data. Security is maintained as long as the number of non-corrupt computational parties is above a certain predetermined threshold.

Different cryptographic techniques can be used to realize SMC. Three common underlying techniques for SMC protocols are 1) homomorphic encryption, 2) garbled circuit evaluations, and 3) linear secret sharing. In this tutorial, we focus on garbled circuit evaluations and secret sharing commonly used for the two-party and multi-party settings, respectively, which do no rely on public-key cryptography.

An SMC protocol is expected to be shown secure against a formal security definition specifying the adversarial model. The two most fundamental and now standard security models correspond to modeling the computation participants as semi-honest (also called honest-but-curious or passive) or malicious (also called active). A semi-honest participant is trusted to follow the prescribed computation, but might save and analyze intermediate results in the attempt to learn unauthorized information, while a malicious participant can arbitrarily deviate from the protocol specification in the attempt to breach security.

During recent years, SMC techniques have experienced dramatic advances in their performance, which applies to both garbled circuit evaluation and computation based on secret sharing in the semi-honest and malicious adversarial models. For that reason, we find it important to include recent advances in the tutorial. At high level, we start by giving the problem definition and describing the

security models. For each type of the technique, we then continue with the fundamental techniques and follow with recent advances and optimizations in both adversarial models. In the following, we briefly describe the two types of the techniques and their recent advances that we intend to cover.

1.1 Garbled Circuit Evaluation

Garbled circuit evaluation (GCE) allows two parties, P_1 and P_2, to securely evaluate a Boolean circuit of their choice. That is, given an arbitrary function $f(x_1, x_2)$ that depends on private inputs x_1 and x_2 of P_1 and P_2, respectively, the parties first represent it as a Boolean circuit. One party, say P_1, acts as a circuit generator and creates a garbled representation of the circuit by associating each of the two values of each binary wire with a random label. The other party, say P_2, acts as a circuit evaluator and evaluates the circuit in its garbled representation without knowing the meaning of the labels that it handles during the evaluation. The output labels can be mapped to their meaning and revealed to either or both parties.

An important component of GCE is a 1-out-of-2 Oblivious Transfer (OT). It allows the circuit evaluator to obtain wire labels corresponding to its private inputs. In particular, in OT the sender (who is the circuit generator in our case) possesses two strings s_0 and s_1 and the receiver (the circuit evaluator) has a bit σ. OT allows the receiver to obtain string s_σ, while the sender learns nothing. When the number of inputs is large, an important optimization is the use of an oblivious transfer extension which allows any number of OTs to be realized using a constant number of regular OT protocols with small additional overhead per input bit. The literature contains many realizations of OT and its extensions (e.g., [2, 16, 26] and others), but in this tutorial we primarily focus on recent efficient constructions [2, 3, 26].

We start coverage of GCE with the original Yao construction [30] for semi-honest users which has conceptual simplicity and gradually bring it closer to the form used today. We plan to discuss modern performance improvement including 1) the free XOR technique [20], which allows XOR gates to be evaluated very cheaply; 2) the garbled row reduction technique [27], which reduces the size of garbled gates; 3) the half-gates optimization [32], which further reduces the size of garbled gates; and 4) performing garbling in a way to permit the use of hardware AES instructions [4], greatly improving the speed of garbling and evaluation.

After the advent of the original GCE [30], several solutions for making it resilient to malicious behavior have been developed [13, 14, 22, 23, 29]. In this tutorial, we cover the cut-and-choose technique [22, 23, 29], as a popular approach to making SFE based on GCE resilient to malicious behavior. However, the cut-and-choose technique alone does not provide full security in the presence of malicious participants, and additional attacks can be mounted. The attacks include 1) providing inconsistent inputs into multiple circuits by the garbler or evaluator [21, 23, 33], 2) performing a selective failure attack [19, 23, 25], and 3) performing an output authenticity attack [18, 21, 23, 29]. In this tutorial, we cover these attacks and their defenses.

1.2 Secret Sharing

Secret sharing allows private values to be split into random shares, which are distributed among a number of parties, and perform computation directly on secret-shared values without computationally-expensive cryptographic operations. In an (n, t)-secret sharing scheme [5, 28], any private value is secret-shared among n parties such that any $t + 1$ (or more) shares can be used to reconstruct the secret, while t or fewer share holders cannot learn any information about the shared value, i.e., it is protected in the information-theoretic sense. In a linear secret sharing (LSS) scheme (e.g., [28]), a linear combination of secret-shared values can be performed by each party on its shares locally, without any interaction, but multiplication of secret-shared values requires communication between all of them. Based on the relationship between n and t, secret sharing based approaches can be divided into two categories: 1) those with honest majority and 2) those without. Based on the underlying construction and adversarial model, different thresholds t can be defined for the number of corrupt computational parties in the categories with honest majority and without honest majority. For example, for constructions with honest majority, it is common to have $t < n/2$ in the semi-honest model and $t < n/3$ in the malicious model.

Security of SMC protocols based on secret sharing for semi-honest parties can be extended to the malicious security model. In that case, to show security of a protocol in the presence of malicious adversaries, we need to generally ensure that all participants follow the steps of the computation. Traditionally this have been achieved via each party proving that each step of their computation was performed correctly using verifiable secret sharing, while more recent constructions may deviate from this mechanism. Additional proofs associated with this setting include proofs that shares of a private value were distributed correctly among the participants (when the dealer is dishonest) and proofs of proper reconstruction of a value from its shares (when not already implied by other techniques). A significant difference between constructions with and without honest majority is with respect to computation completion. In the honest majority setting, if at most t dishonest participants quit, others are able to reconstruct their shares and proceed with the rest of the computation, while in the setting without honest majority deviation from the computation by corrupt parties leads to aborting the computation. A large number of publications provide constructions based on LSS secure in the malicious model (e.g., [1, 8, 12] and many others).

For the secret sharing part of this tutorial, we plan to cover a number of standard and more recent secret sharing solutions in the presence of semi-honest and malicious participants. We first introduce standard secret sharing constructions for the semi-honest adversarial model [5, 28]. We then proceed with general and efficient protocols proposed in [10] which work for both semi-honest and malicious models with honest majority. Consequently, we discuss SPDZ [11], which is a recent construction secure in the presence of malicious users with up to $n - 1$ corrupt parties (i.e., no honest majority), and its further improvements such as [9, 17]. The solution has a very efficient online phase for a construction secure in the malicious model and a pre-processing phase based on homomorphic encryption.

We conclude the tutorial by discussing a number of compilers for secure two- and multi-party computation that use GCE and LSS such as CBMC-GC [15], the compiler in [21], ObliVM [24], Obliv-C [31], Sharemind [6], VIFF [7], PICCO [34], and SPDZ's followup. We also comment on them in terms of the supported setting and adversarial model, efficiency, and functionality.

REFERENCES

[1] G. Asharov, Y. Lindell, and T. Rabin. 2011. Perfectly-Secure Multiplication for Any $t < n/3$. In *CRYPTO*. 240–258.

[2] G. Asharov, Y. Lindell, T. Schneider, and M. Zohner. 2013. More Efficient Oblivious Transfer and Extensions for Faster Secure Computation. In *ACM Conference on Computer and Communications Security (CCS)*. 535–548.

[3] G. Asharov, Y. Lindell, T. Schneider, and M. Zohner. 2015. More Efficient Oblivious Transfer Extensions with Security for Malicious Adversaries. In *EUROCRYPT*. 673–701.

[4] M. Bellare, V. Hoang, S. Keelveedhi, and P. Rogaway. 2013. Efficient Garbling from a Fixed-Key Blockcipher. In *IEEE Symposium on Security and Privacy (S&P)*. 478–492.

[5] G. R. Blakley. 1979. Safeguarding Cryptographic Keys. In *National Computer Conference*, Vol. 48. 313–317.

[6] D. Bogdanov, S. Laur, and J. Willemson. 2008. Sharemind: A Framework for Fast Privacy-Preserving Computations. In *European Symposium on Research in Computer Security (ESORICS)*. 192–206.

[7] I. Damgård, M. Geisler, M. Krøigaard, and J. B. Nielsen. 2009. Asynchronous Multiparty Computation: Theory and Implementation. In *International Workshop on Public Key Cryptography (PKC)*. 160–179.

[8] I. Damgård, Y. Ishai, and M. Krøigaard. 2010. Perfectly Secure Multiparty Computation and the Computational Overhead of Cryptography. In *EUROCRYPT*. 445–465.

[9] I. Damgård, M. Keller, E. Larraia, V. Pastro, P. Scholl, and N. P. Smart. 2013. Practical Covertly Secure MPC for Dishonest Majority–Or: Breaking the SPDZ Limits. In *European Symposium on Research in Computer Security (ESORICS)*. 1–18.

[10] I. Damgård and J. B. Nielsen. 2007. Scalable and Unconditionally Secure Multiparty Computation. In *CRYPTO*. 572–590.

[11] I. Damgård, V. Pastro, N. Smart, and S. Zakarias. 2012. Multiparty Computation from Somewhat Homomorphic Encryption. In *CRYPTO*. 643–662.

[12] R. Gennaro, M. Rabin, and T. Rabin. 1998. Simplified VSS and Fast-Track Multiparty Computations with Applications to Threshold Cryptography. In *ACM Symposium on Principles of Distributed Computing (PODC)*. 101–111.

[13] O. Goldreich, S. Micali, and A. Wigderson. 1991. Proofs that Yield Nothing but their Validity or All Languages in NP Have Zero-Knowledge Proof Systems. *J. ACM* 38, 3 (1991), 690–728.

[14] S. Goldwasser, S. Micali, and A. Wigderson. 1987. How to Play Any Mental Game, or a Completeness Theorem for Protocols with an Honest Majority. In *ACM Symposium on the Theory of Computing (STOC)*. 218–229.

[15] A. Holzer, M. Franz, S. Katzenbeisser, and H. Veith. 2012. Secure Two-Party Computations in ANSI C. In *AMC Conference on Computer and Communications Security (CCS)*. 772–783.

[16] Y. Ishai, J. Kilian, K. Nissim, and E. Petrank. 2003. Extending Oblivious Transfers Efficiently. In *CRYPTO*. 145–161.

[17] M. Keller, V. Pastro, and D. Rotaru. 2018. Overdrive: Making SPDZ Great Again. In *EUROCRYPT*. 158–189.

[18] M. Kiraz. 2008. *Secure and Fair Two-Party Computation*. Ph.D. Dissertation. Technische Universiteit Eindhoven.

[19] M. Kiraz and B. Schoenmakers. 2006. A Protocol Issue for the Malicious Case of Yao's Garbled Circuit Construction. In *Symposium on Information Theory in the Benelux*. 283–290.

[20] V. Kolesnikov and T. Schneider. 2008. Improved Garbled Circuit: Free XOR Gates and Applications. In *International Colloquium on Automata, Languages, and Programming (ICALP)*. 486–498.

[21] B. Kreuter, A. Shelat, and C. H. Shen. 2012. Billion-Gate Secure Computation with Malicious Adversaries. In *USENIX Security Symposium*. 285–300.

[22] Y. Lindell. 2013. Fast Cut-and-Choose Based Protocols for Malicious and Covert Adversaries. In *CRYPTO*. 1–17.

[23] Y. Lindell and B. Pinkas. 2007. An Efficient Protocol for Secure Two-Party Computation in the Presence of Malicious Adversaries. In *EUROCRYPT*. 52–78.

[24] C. Liu, X. S. Wang, K. Nayak, Y. Huang, and E. Shi. 2015. ObliVM: A Programming Framework for Secure Computation. In *IEEE Symposium on Security and Privacy (S&P)*. 359–376.

[25] P. Mohassel and M. Franklin. 2006. Efficiency Tradeoffs for Malicious Two-Party Computation. In *International Workshop on Public Key Cryptography (PKC)*. 458–473.

[26] M. Naor and B. Pinkas. 2001. Efficient Oblivious Transfer Protocols. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 448–457.

[27] B. Pinkas, T. Schneider, N. P. Smart, and S. C. Williams. 2009. Secure Two-Party Computation is Practical. In *ASIACRYPT*. 250–267.

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

[29] A. Shelat and C. Shen. 2011. Two-Output Secure Computation with Malicious Adversaries. In *EUROCRYPT*. 386–405.

[30] A. C. Yao. 1986. How to Generate and Exchange Secrets. In *IEEE Symposium on Foundations of Computer Science (FOCS)*. 162–167.

[31] S. Zahur and D. Evans. 2015. Obliv-C: A Language for Extensible Data-Oblivious Computation. IACR Cryptology ePrint Archive Report 2015/1153. (2015).

[32] S. Zahur, M. Rosulek, and D. Evans. 2015. Two Halves Make a Whole: Reducing Data Transfer in Garbled Circuits Using Half Gates. In *EUROCRYPT*. 220–250.

[33] Y. Zhang, M. Blanton, and F. Bayatbabolghani. 2017. Enforcing Input Correctness via Certification in Garbled Circuit Evaluation. In *European Symposium on Research in Computer Security (ESORICS)*. 552–569.

[34] Y. Zhang, A. Steele, and M. Blanton. 2013. PICCO: A General-Purpose Compiler for Private Distributed Computation. In *ACM Conference on Computer and Communications Security (CCS)*. 813–826.

BIOGRAPHY OF THE SPEAKERS

Fattaneh Bayatbabolghani is a postdoctoral scholar in the School of Information at the University of California, Berkeley. Before joining Berkeley, she worked in the School of Informatics, Computing, and Engineering at Indiana University-Bloomington as a postdoctoral fellow. She received her PhD in Computer Science and Engineering from the University of Notre Dame in 2017 and MS in Computer Science and Engineering from the University of Notre Dame in 2016. Her primary research interest is in designing general and efficient privacy-preserving solutions motivated by real-world applications. During her PhD, she designed novel, general, and efficient privacy-preserving solutions and protocols which can be used for different applications, but are specifically used for biometric computations. Her research was published in competitive outlets including PETS, IEEE S&P magazine, IJIS, ESORICS, and GenoPri. She organized an interactive workshop on the human aspects of smarthome security and privacy at SOUPS'18 (Co-located with USENIX Security'18). She also serves as a reviewer for venues in information security such as the IEEE Transactions on Information Forensics and Security. Email:fbayatba@ischool.berkeley.edu

Marina Blanton is an Associate Professor in the Department of Computer Science and Engineering at the University at Buffalo, The State University of New York. She received her PhD in Computer Science from Purdue University in 2007, MS in Computer Science from Purdue University in 2004, and MS in EECS from Ohio University in 2002. Her research interests lie in applied cryptography, security, and privacy and currently primarily focus on secure computation and outsourcing. Dr. Blanton has over 70 research publications in competitive venues and has won multiple awards for her research, including an AFOSR Young Investigator Award in 2013 and the 2015 CCS Test of Time Award. Dr. Blanton is actively involved in the community service, currently serving as an Associate Editor for the IEEE Transactions on Information Forensics and Security and on many conference program committees including CCS, Oakland, and NDSS. She is a senior member of both ACM and IEEE. Email: mblanton@buffalo.edu