ARC'18
مؤتمر مؤسسة قطر السنوي للبحوث
QATAR FOUNDATION ANNUAL RESEARCH CONFERENCE
البحث والتطوير: التركيز على الأولويات، وإحداث الأثر
R&D: FOCUSING ON PRIORITIES, DELIVERING IMPACT
19-20 مـــارس
19-20 MARCH

مـؤسـسـة قـطـر
Qatar Foundation
لإطـلاق قـدرات الإنـسـان.
Unlocking human potential.

Computing & Information Technology - Poster Display

http://doi.org/10.5339/qfarc.2018.ICTPD610

Enabling Efficient Secure Multiparty Computation Development in ANSI C

Ahmad Musleh*, Soha Hussein, Khaled M. Khan, Qutaibah M. Malluhi

Qatar University
* ahmad.s.musleh@gmail.com

Secure Multi-Party Computation (SMPC) enables parties to compute a pub- lic function over private inputs. A classical example is the millionaires problem, where two millionaires want to figure out who is wealthier without revealing their actual wealth to each other. The insight gained from the secure computa- tion is nothing more than what is revealed by the output (in this case, who was wealthier but not the actual value of the wealth). Other applications of secure computation include secure voting, on-line bidding and privacy-preserving cloud computations, to name a few. Technological advancements are making secure computations practical, and recent optimizations have made dramatic improve- ments on their performance. However, there is still a need for effective tools that facilitate the development of SMPC applications using standard and famil- iar programming languages and techniques, without requiring the involvement of security experts with special training and background. This work addresses the latter problem by enabling SMPC application de- velopment through programs (or repurposing existing code) written in a stan- dard programming language such as ANSI C. Several high-level language (HLL) platforms have been proposed to enable secure computation such as Obliv-C [1], ObliVM [2] and Frigate [3] These platforms utilize a variation of Yao's garbled circuits [4] in order to evaluate the program securely. The source code written for these frameworks is then converted into a lower-level intermediate language that utilizes garbled circuits for program evaluation. Garbled Circuits have one party (garbler) who compiles the program that the other party (evaluator) runs, and the communication between the two parties happens through oblivi- ous transfer. Garbled circuits allow two parties to do this evaluation without a need for a

دار جامعة حمد بن خليفة للنشر
HAMAD BIN KHALIFA UNIVERSITY PRESS

Cite this article as: Musleh A et al. (2018). Enabling Efficient Secure Multiparty Computation Development in ANSI C. Qatar Foundation Annual Research Conference Proceedings 2018: ICTPD610 http://doi.org/10.5339/qfarc.2018.ICTPD610.

trusted third party. These frameworks have two common characteristics: either define a new language [2] or make a restricted extension of a current language [1]. This is somewhat prohibitive as it requires the programmer to have a sufficient under- standing of SMPCs related constructs and semantics. This process is error-prone and time-consuming for the programmer. The other characteristic is that they use combinational circuits, which often require creating and materializing the entire circuit (circuit size may be huge) before evaluation. This introduces a restriction on the program being written. TinyGarble [5], however, is a secure two-party computation framework that is based on sequential circuits. Compared with the frameworks mentioned earlier, TinyGarble outperforms them by orders of magnitude. We are developing a framework that can automatically convert a HLL pro- gram (in this case ANSI C) into an hardware definition language, which is then evaluated securely. The benefit of having such transformation is that it does not require knowledge of unfamiliar SMPC constructs and semantics, and per- forms the computation in a much more efficient manner. We are combining the efficiency of sequential circuits for computation as well as the expressiveness of a HLL like ANSI C to be able to develop a secure computation framework that is expected to be effective and efficient. Our proposed approach is two-fold: first, it offers a separation of concern between the function of computation, written in C, and a secure computation policy to be enforced. This leaves the original source code unchanged, and the programmer is only required to specify a policy file where he/she specifies the function/variables which need secure computations. Secondly, it leverages the current state-of-the-art framework to generate sequential circuits. The idea is to covert the original source code to Verilog (a Hardware Definition Language) as this can then be transformed into standard circuit description which TinyGarble [5] would run. This will enable us to leverage TinyGarbles efficient sequential circuits. The result would be having the best of both worlds where we have HLL that would be converted and evaluated as a sequential circuit. References [1] S. Zahur and D. Evans, “Obliv-c: A language for extensible data-oblivious computation,” IACR Cryptology ePrint Archive, vol. 2015, p. 1153, 2015. [2] C. Liu, X. S. Wang, K. Nayak, Y. Huang, and E. Shi, “Oblivm: A pro- gramming framework for secure computation,” in 2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015, pp. 359–376, 2015. [3] B. Mood, D. Gupta, H. Carter, K. R. B. Butler, and P. Traynor, “Frigate: A validated, extensible, and efficient compiler and interpreter for secure com- putation,” in IEEE European Symposium on Security and Privacy, EuroS&P 2016, Saarbru¨ cken, Germany, March 21-24, 2016, pp. 112–127, 2016. [4] A. C. Yao, “Protocols for secure computations (extended abstract),” in 23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982, pp. 160–164, 1982. [5] E. M. Songhori, S. U. Hussain, A. Sadeghi, T. Schneider, and F. Koushanfar, “Tinygarble: Highly compressed and scalable sequential garbled circuits,” in 2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015, pp. 411–428, 2015.