CS-523 SMCompiler Report

Zoé Marin, Theresa Tratzmüller

Abstract—**We present an implementation of a secure multi-party computation infrastructure which allows a group of participants to perform arithmetic operations on their joint inputs without revealing them. The implementation is evaluated in terms of the incurred computation- and communication cost. An example use case is described to illustrate the applicability of the system.**

I. Introduction

Secure multi-party computation (SMC) is a cryptographic technique that enables the privacy-preserving distributed evaluation of boolean or arithmetic circuits in settings involving two or more participants [1]. Areas of application are (among many others) secure auctions, secure electronic voting and secure machine learning [2]. In this work, we demonstrate a system that allows an arbitrary number of participants, each in possession a numer of secret inputs, to jointly compute the value of an arithmetic expression. For expressions involving the multiplication of secret inputs, our system relies on a trusted third party. We present a thorough examination of the system's performance as well as a sample use case, where SMC is utilized to enable academic collaboration in the absence of mutual trust.

II. Threat model

The policy to be maintained in the context of SMC protocols is that the private inputs of the participants remain private: the participants may learn nothing but the result of the computation [1]. We assume secret and authenticated channels between parties, rendering potential network eavesdroppers out of scope of our threat model. The third party required for the evaluation of arithmetic expressions involving the multiplication of secret inputs is trusted and therefore also out of scope. Hence, the only adversaries from the point of view of a participant are the other participants. We assume the adversaries to be semi-honest (passive), meaning that they follow the protocol, but are interested in breaking the privacy or their fellow participants [1]. Given additive secret sharing among n participants, where participants divide their secrets into n shares so that recovering the secret requires summing all of the shares, it is indeed necessary to corrupt all of the participants to be able to learn the secret's value. Note however that in the case of some arithmetic expressions, with the prime example being a simple addition of one secret value per participant, $n-1$ colluding parties may be able to jointly recover the secret input of the remaining one non-colluding party.

III. Implementation details

This implementation supports computing arithmetic expression composed of addition, subtraction and multiplication operations. The two operands in each of these operations can be composed of secret values belonging to the individual participants, scalars (which are not secret), or both. Cryptographically, the ability to carry out these computations without the participants having to reveal their secrets is achieved by means of an additive secret sharing scheme (i.e., each participant divides each of their secrets into as many shares as there are participants). Participants compute the value of an expression based on the shares they have access to. In the end, the final result of the computation is reconstructed from the individual participants' local computation results. For multiplication of operands containing secrets, the *Beaver Triplet Scheme* is employed [3]. For the provision of the blinding values required by the scheme, we introduce a trusted third party.

IV. Performance evaluation

An evaluation of computation- and communication cost was carried out on a machine with a 1.2 GHz Quad-Core Intel Core i7 processor, using one core. With respect to the SMC parties participating in the computation of an arithmetic circuit, computation time was measured in the form of the total runtime as well as the time taken for sharing secrets, processing expressions as well as reconstructing secrets. Communication cost incurred for the SMC parties was assessed in the form of the number of bytes sent and received over the network. Regarding the trusted third party (TTP), computation time was measured based on the time taken to generate beaver triplets and shares thereof. Communication cost in the case of the TTP was measured in the form of bytes sent (the TTP is only contacted via GET requests in the protocol). All measurements of time were corrected for network delays. We systematically varied the number of SMC parties, the number of secret additions, the number of scalar additions, the number of secret multiplications, as well as the number of scalar multiplications (while keeping all other factors constant in each case). For each concrete value of the factors varied, the computation was repeated 30 times to allow centrality trends to show up with numeric significance. The results of the performance evaluation are displayed in 1. Note that in each experiment, only those metrics that could reasonably be expected to be influenced by the varied factor were considered.

V. Application

Our sample application deals with scientific collaboration in the absence of mutual trust. The setting is the following: Researchers at the *École fantastique de Gruyère (EFG)* have recently developed a psychological scale called the *Good Guy Inventory (GGI)*. It reliably tells saints from villains and

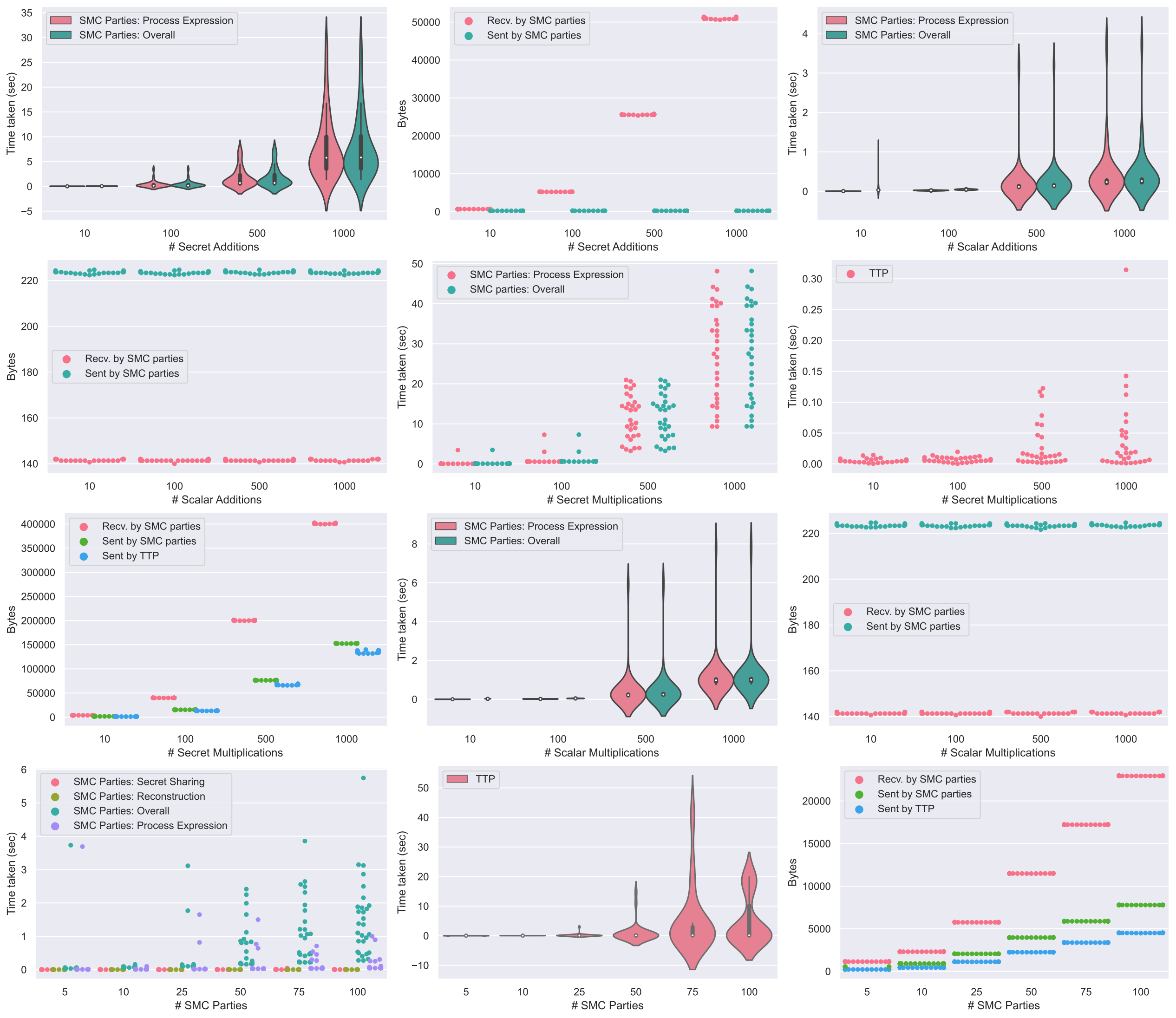

Fig. 1: Computation and communication cost evaluation results

exhibits excellent statistical properties. In the light of increasing awareness for sustainability and corporate ethics, *Bestlé S.A.* wonders if their CEO is a good person that represents their company well. Hence, they demand that the CEO be assessed using the *GGI*. A problem however presents itself in the form of the traditional hostility between the left-leaning academics of the *EFG* and the *Bestlé* CEO. The assessment must therefore be performed by researchers at the *Hochschule St. Quallen (HSQ)*, who are in favor of multinational corporations. The values obtained for the individual dimensions of the questionnaire in this assessment may not be revealed to the team from *EFG*. The researchers from *EFG* agree to provide their scale, however, they do not want to reveal the weights given to the individual facets of the questionnaire which are top-secret. Thus, they decide to use SMC to compute the CEO's score, so that none of the two teams of researchers must disclose the data they have access to. Further, from the score shall be subtracted the corruptness-index the canton of Vaud has determined for *Bestlé* for the past year. This score is public information. Finally, the result is multiplied with a scalar for scaling. With the third party being trusted and communication channels being assumed secure, the two research teams are each other's sole adversaries in this scenario. As both of them are interested in following the protocol (press coverage about the collaboration will benefit both of them) but curious about the other's inputs, we are in a honest-but-curious scenario. A privacy leakage may be present if the CEO achieved a score of 0 on all the subscales of the *GGI*. This could be mitigated by making the corruptness index secret, which would however violate the canton of Vaud's transparency standard.

REFERENCES

[1] N. P. Smart *et al.*, *Cryptography: an introduction*. McGraw-Hill New York, 2003, vol. 3.

[2] D. Evans, V. Kolesnikov, M. Rosulek *et al.*, "A pragmatic introduction to secure multi-party computation," *Foundations and Trends® in Privacy and Security*, vol. 2, no. 2-3, pp. 70–246, 2018.

[3] D. Beaver, "Efficient multiparty protocols using circuit randomization," in *Annual International Cryptology Conference*. Springer, 1991, pp. 420–432.