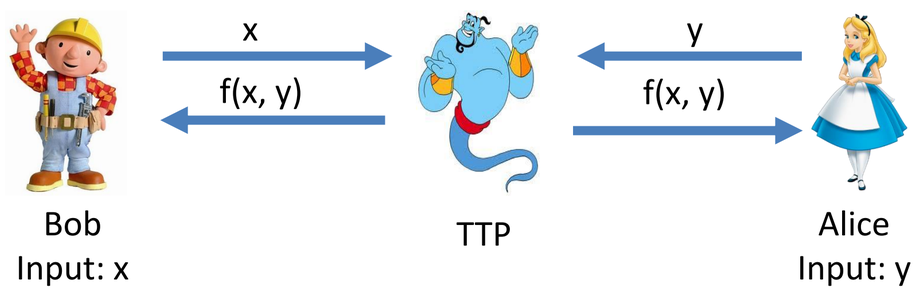
Computing on Private Data in High Latency Networks

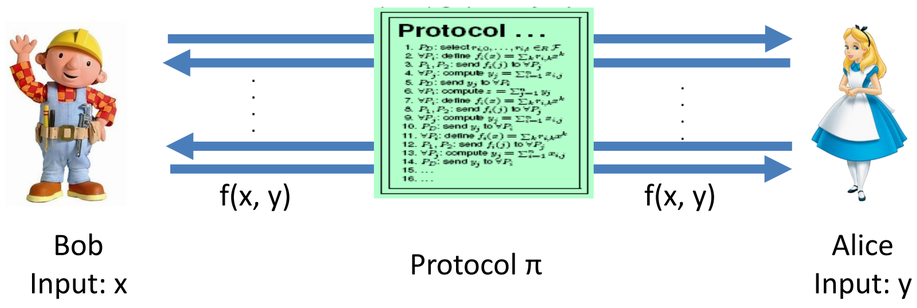
Arpita Patra and Pratik Sarkar
Department of Computer Science & Automation, Indian Institute of Science

Introduction

Suppose two parties Bob and Alice want to compute any arbitrary function f(x, y) on their private inputs x and y respectively s.t. Bob (resp. Alice) gets no information about y (resp. x) other than the output. Ideally, this can be performed using a Trusted Third party (TTP).



However, TTPs do not exist in real life. Hence, we rely on **Secure Two-Party Computation (2PC)** to perform this task**.** Secure 2PC allows Bob and Alice to compute any arbitrary function f(x, y) on their inputs x and y by executing a protocol π.

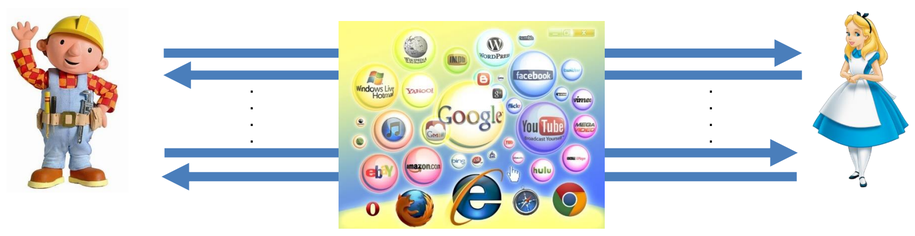


π has to satisfy the following two properties:
Correctness: The following should hold f(x, y) = π(x, y) for all x, y pairs.
Security: If Adversary 𝔸 corrupts Bob, then he obtains no more information about y, from protocol π, than what can be computed from f(x, y). Same holds when 𝔸 corrupts Alice and Bob is honest.

Our Adversarial Model:
Static - 𝔸 corrupts one (or both) parties at the outset of the protocol.
Malicious - 𝔸 can arbitrarily deviate from the protocol steps and follow his own adversarial algorithm, while controlling a corrupted party.

Motivation

We consider the problem of Secure 2PC between Alice and Bob in a setting, where the parties are connected through **high-latency networks**, like the Internet. In high latency networks, sending messages causes delay in the network.



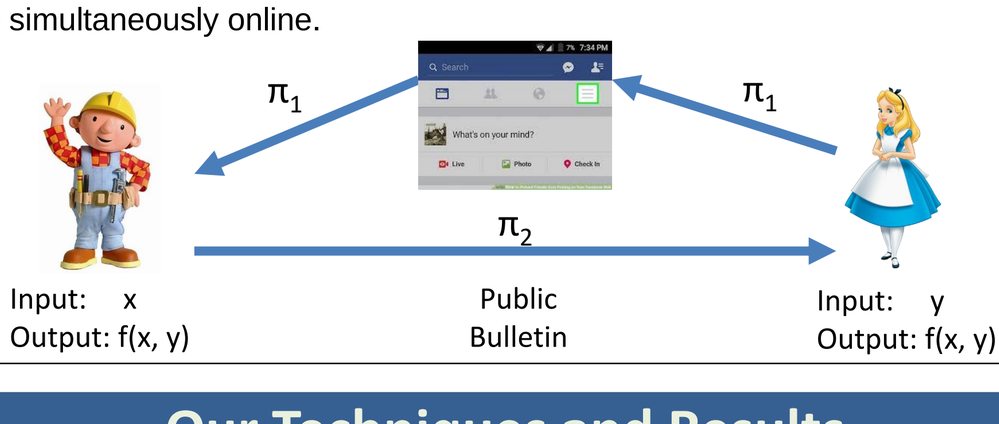
High Latency Networks

It models practical situations where the parties are connected through slow connections. In such cases, sequential message exchanges between the parties increases the round complexity of the protocol; thereby drastically increasing the delay and affecting the runtime of the protocol.

In order to address this issue, we try to reduce the interaction between the parties, by keeping the round complexity of the protocol at a minimum.

Our Approach

Our protocol π requires 1 interaction, i.e. **2 rounds** of message communication between Alice and Bob. This is round-optimal since 2 rounds is necessary [MW16] for Secure 2PC. We denote $\pi = (\pi_1, \pi_2)$, where π_1 and π_2 is the first and second message of π. π can be considered **Non - Interactive** since Alice can post π_1 on a public bulletin, e.g. her Facebook Wall. Later, Bob read π_1 when he is online, and complete the protocol by sending π_2 to Alice. Both parties need not be simultaneously online.



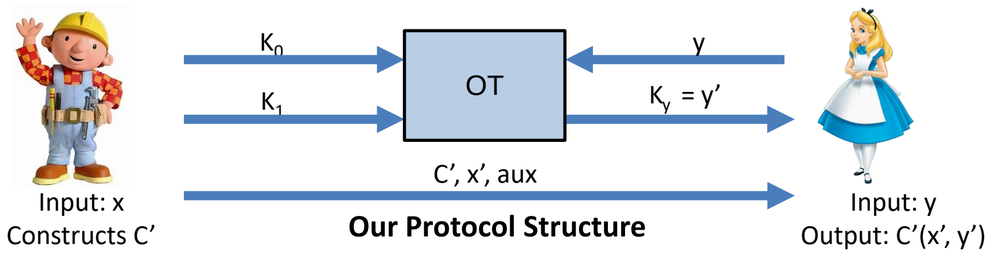
Our Techniques and Results

We use the primitives of Yao's Garbled Circuit (GC) and Oblivious Transfer (OT) to construct our Secure 2PC protocol π.

Garbled Circuit: It is an encrypted representation, C', of a circuit C, computing the function f(x, y). It is constructed by a party known as constructor. Another party known as evaluator can evaluate C' given the garbled inputs, x' and y', corresponding to x and y respectively. Correctness holds when C'(x', y') = f(x, y) and it is secure when x or y are not leaked to evaluator or constructor respectively.

Oblivious Transfer: It is a protocol between two parties: a sender S and a receiver R, where S holds a pair of strings (m_0, m_1) and R holds a selection bit b. At the end of the protocol, R learns m_b only, S gets no information about b.

Next, we give a high level overview of our protocol π as follows. For simplicity we assume that x and y are bits.



In the diagram, C' are the garbled circuits, x' and y' are the garbled inputs and aux contains auxiliary information to ensure security against a malicious 𝔸, who can corrupt Alice or Bob or both. The OT is performed in 2 rounds, where Alice sends the first message; thereby fixing the round complexity to 2. Bob can also obtain the output in 2 rounds by symmetric execution of the protocol, with some changes in π.

Current Results: We have constructed a 2 round OT [BPRS17] which can be efficiently implemented using hash functions and few public key operations. We reduced the size of aux and improved the runtime of the protocol over previous results [IKOPS11,AMPR14, MR17]. We have also considered the case where multiple executions of f is performed. In such a case, the runtime and communication for a single run gets amortized and it is more optimized, when compared with a single execution of f performed in isolation.

Ongoing Work: We are developing more efficient constructions for C' which would significantly reduce the communication and runtime of the protocol. Another optimization under consideration is to make the first message of OT independent of |y|. Currently, the first OT message is linearly dependent on |y|. This optimization would allow Alice to broadcast few (security parameter size) bits, as the first message π_1 on the public domain even if y is arbitrarily large.

Contact

Pratik Sarkar
Department of Computer Science & Automation
Indian Institute of Science
Email: pratiks@iisc.ac.in
Phone: 08277359870

References

1. [AMPR14] Arash Afshar, Payman Mohassel, Benny Pinkas and Ben Riva. *Non-Interactive Secure Computation Based on Cut-and-Choose.* Eurocrypt 2014, 387-404
2. [BPRS17] Megha Byali, Arpita Patra, Divya Ravi and Pratik Sarkar. *Efficient, Round-optimal, Universally-Composable Oblivious Transfer and Commitment Scheme with Adaptive Security.* IACR Cryptology ePrint Archive 2017: 1165 (2017)
3. [IKOPS11] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, Manoj Prabhakaran and Amit Sahai. *Efficient Non-interactive Secure Computation*. Eurocrypt 2011, 406-425
4. [MR17] Payman Mohassel and Mike Rosulek, *Non-interactive Secure 2PC in the Offline/Online and Batch Settings.* In Eurocrypt 2017, pages 425-455
5. [MW16] Pratyay Mukherjee and Daniel Wichs. *Two Round Multiparty Computation via Multi-Key FHE.* Eurocrypt 2016, 735-763.