622.755 – Introduction to Cybersecurity Summer Term, 2022/23

Project report on Yao's Protocol for MPC

Lorenzo Bazzana 12245823

1 Outline

This report presents a description on the security aspects of Yao's Garbled Circuit protocol [9], together with a brief dissection on real world implementations of the protocol and MPC in general, and a reflection on the Social, Ethical and Legal aspects that are involved in the use of MPC.

Section 2 presents Yao's protocol and other Multi-Party Computation variants, while section 3 describes some of the real life uses of MPC techniques. Section 4 deals with the SEL aspects related to the use of MPC and its consequences on everyday life.

2 Multi-Party Computation

2.1 Yao's Protocol

Yao's Garbled Circuit protocol is a cryptographic protocol for two-party secure computation, used to compute the result of a function $f(x_1, ..., x_n, y_1, ..., y_m)$ over two sets of inputs $X = \{x_1, ..., x_n\}$ and $Y = \{y_1, ..., y_m\}$ owned by two different parties (Alice and Bob), in a way such that at the end of the protocol both parties know the final result but nothing else about the other party's inputs, with the exception of what can be interpreted by the output.

The protocol uses *garbled circuits*, meaning that the function f is first represented as a boolean circuit, and then it is evaluated over encrypted versions of the gates; this allows to compute the output without revealing the input. Since its first formulation, many improvements to the protocol have been proposed [3, 6, 2], such as the *free-XOR* [6] technique.

2.2 Protocol implementation

A toy example of a function that can be computed using Yao's protocol is the maximum value in the set of integer values owned by Alice and Bob:

$$f(X, Y) = \max_{v \in (X \cup Y)}(v)$$

Since the goal is to implement a secure computation that does not reveal any information on Alice's and Bob's inputs except for the result, the simplest choice is to set up the computation with the following two steps:

1. Alice and Bob both compute locally their own maximum values a_{max} and b_{max}
2. The global maximum is computed using Yao's protocol, with inputs a_{max} and b_{max}. At the end of the computation both Alice and Bob know such value.

On a high-level approach, we would like to guarantee that the result is *correct* (i.e. the protocol actually computes the function f previously described) and that the computation is *private* (i.e. nothing is learned from the protocol other than what can be deduced from the output) [7].

2.3 Other MPC techniques

Other examples of techniques used for function computation using private data are *secret sharing* and *homomorphic encryption*.

Secret sharing is an MPC method that consists in dividing a secret in different parts (shares) and distributing each share to a different party, so that no party has a complete knowledge of other parties' secrets; the shares can then be used to compute a function (e.g. the sum or the average of all the secrets) [8].

Homomorphic encryption allows to compute functions on private encrypted data without having to decrypt it. Although homomorphic encryption can be considered as independent from MPC[1], it can also be used for MPC purposes: for example, [5] presents a way to compute private set intersections using homomorphic encryption.

3 Applications of MPC

Multi-Party Computation can be applied in real world *data analysis* scenarios where data privacy is required (for example, privacy-preserving machine learning and secure genomic sequence comparison [10]).

MPC has only recently seen large scale practical applications: the first relevant use of MPC for commercial applications was in the 2008 beet auction in Denmark between the nation's beet sugar producer "Danisco" and Danish farmers [4]. From there many other practical uses of MPC have emerged; some of those will be presented in the following.

3.1 MPC wallets for Blockchain applications

MPC has been employed in blockchain private key management in the so called "MPC wallets" such as "ZenGo"[2] and "Coinbase"[3], which make use of MPC to divide private keys among different actors, with the aim to avoid key theft and removing single points

[1]This is because it can be used to outsource computation to a third party, using just our own encrypted data.

[2]ZenGo: https://zengo.com

[3]Coinbase: https://www.coinbase.com/

of failure (SPOF) in key management: even if a single part of a key distributed among n parties is compromised, the relative wallet is still safe from attacks, since all shares are required to sign a transaction, and MPC signatures can only be computed via authentication of the parties.

3.2 Private databases

Another application of MPC techniques is Private Data as a Service (PDaaS): privacy is an ever important requirement for data management, and for this reason being able to perform operations on data without revealing or decrypting it makes MPC the prominent tool in the field.

An example of such PDaaS is Galois, inc.'s "Jana", a distributed and scalable database model where every SQL operation is performed on encrypted data, using different kinds of encryption that allow data comparison in some form, even though this implies a small amount of information leakage (a trade that is necessary to maintain general privacy) [1].

4 SEL considerations

4.1 Advantages of MPC to Society

The large scale applications of MPC could greatly benefit society as a whole: the world is moving in a direction where IoT and the Cloud are more and more important for everyday life applications. Having the possibility of preserving privacy even when computing shared data is crucial in such a world. MPC could also improve fields where massive data is required, like Machine Learning: being able to train Machine Learning models using data from private users whilst maintaining privacy could be very beneficial for the accuracy of the model, because it would be trained on a wide and varied dataset.

Another consequence of large MPC applications is collaboration: data is a valuable resource, and for this reason companies and governments are reluctant to share it; MPC instead allows collaboration without revealing the data, so it could open new ways of collaboration between parties at different levels.

4.2 Ethical implications

MPC's usage in a global setting must guarantee that these techniques are secure even in an adversarial environment. Malicious parties could leverage on the privacy given by MPC for their advantage, be it trying to learn information on other users' data, or using the shared computational power in some other form. For example, multinational companies could use MPC for statistical market analysis using their shared but private data, and using the results to influence the customers or other forms of market manipulation.

From an ethical point of view then these techniques must be used carefully, especially in a context where the legitimacy of the computation results is very important (for example, e-voting). Another aspect involved with ethical and moral questions is data management: MPC brings new ways of treating data, changing the notion of responsibility and accountability of and for that data. Consequently, we must define these concepts in a way that also considers the use of data in MPC, and at the same time ensure that MPC respects those definitions.

4.3 Legal aspects of MPC

The definitions of responsibility and accountability mentioned in the previous paragraph are also deeply connected to the legal aspect of privacy regulations, as in order to guarantee a fair usage of MPC it must comply with all the data and privacy regulations (like the european GDPR). The usage and characteristics of MPC implementations should also be regulated, so that in a commercial setting the users/customers are legally protected against eventual misuse of these technologies (like in the case of some party trying to recover information about the users' data).

Other important legal aspects of data usage that come into play when using MPC are data ownership and intellectual property: even if the data used in MPC is private, it still influences the result of the computation. Because of this it is crucial to clearly define frameworks to evaluate who owns the resulting data, and for what means it can be used.

5 Conclusion

To sum up, in this paper we presented Yao's protocol and other MPC variants, together with some real-world examples of situations where these techniques are actually used. As discussed in section 4, MPC could become more and more important in the future, but there must be careful considerations on its uses and the contexts where it is implemented, as to avoid its misuse and to safeguard users when it comes to the privacy of their data, both on an ethical and a legal perspective.

References

[1] David Archer, Dan Bogdanov, Yehuda Lindell, Liina Kamm, Kurt Nielsen, Jakob Pagter, Nigel Smart, and Rebecca Wright. From keys to databases—real-world applications of secure multi-party computation. *Computer Journal*, 61:1749–1771, 12 2018.

[2] Donald Beaver, Silvio Micali, and Phillip Rogaway. The round complexity of secure protocols. *Proceedings of the twenty-second annual ACM symposium on Theory of Computing*, 1990.

[3] Osman Biçer. Efficiency optimizations on yao's garbled circuits and their practical applications, 2017.

[4] Peter Bogetoft, Dan Lund Christensen, Ivan Damgard, Martin Geisler, Thomas Jakobsen, Mikkel Krøigaard, Janus Dam Nielsen, Jesper Buus Nielsen, Kurt Nielsen, Jakob Pagter, Michael Schwartzbach, and Tomas Toft. Multiparty computation goes live. Cryptology ePrint Archive, Paper 2008/068, 2008. `https://eprint.iacr.org/2008/068`.

[5] Hao Chen, Kim Laine, and Peter Rindal. Fast private set intersection from homomorphic encryption. In *CCS '17 Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1243–1255. ACM New York, NY, USA ©2017, October 2017.

[6] Vladimir Kolesnikov and Thomas Schneider. Improved garbled circuit: Free XOR gates and applications, 07 2008.

[7] Yehuda Lindell and Benny Pinkas. A proof of security of yao's protocol for two-party computation. *Journal of Cryptology*, 22:161–188, 04 2009.

[8] Adi Shamir. How to Share a Secret. *Communications of the ACM*, 22(11):612–613, 1979.

[9] Andrew Chi-Chih Yao. How to generate and exchange secrets. *27th Annual Symposium on Foundations of Computer Science (sfcs 1986)*, pages 162–167, 1986.

[10] Chuan Zhao, Shengnan Zhao, Minghao Zhao, Zhenxiang Chen, Chong-Zhi Gao, Hongwei Li, and Yu an Tan. Secure multi-party computation: Theory, practice and applications. *Information Sciences*, 476:357–372, 2019.