Privacy-Preserving Big Data Analytics: From Theory to Practice

Mohammad G. Raeini and Mehrdad Nojoumian

Department of Computer and Electrical Engineering and Computer Science
Florida Atlantic University, Boca Raton, United States, 33431

Abstract. In the last decade, with the advent of Internet of Things (IoT) and Big Data phenomenons, data security and privacy have become very crucial issues. A significant portion of the problem is due to not utilizing appropriate security and privacy measures in data and computational infrastructures. Secure multiparty computation (secure MPC) is a cryptographic tool that can be used to deal with the mentioned problems. This computational approach has attracted increasing attention, and there has been significant amount of advancement in this domain. In this paper, we review the important theoretical bases and practical advancements of secure multiparty computation. In particular, we briefly review three common cryptographic primitives used in secure MPC and highlight the main arithmetic operations that are performed at the core of secure MPC protocols. We also highlight the strengths and weaknesses of different secure MPC approaches as well as the fundamental challenges in this domain. Moreover, we review and compare the state-of-the-art secure MPC tools that can be used for addressing security and privacy challenges in the IoT and big data analytics. Using secure MPC in the IoT and big data domains is a challenging task and requires significant expert knowledge. This technical review aims at instilling in the reader an enhanced understanding of different approaches in applying secure MPC techniques to the IoT and big data analytics.

Keywords: Secure Multiparty Computation; Secure MPC; Internet of Things (IoT); Big Data Analytics; Yao's Garbled Circuits; Yao's Millionaires Problem; Secret Sharing; Homomorphic Encryption

1 Introduction

In recent years, data privacy has been a critical issue, e.g., the representatives of companies such as Google and Facebook have recently been questioned on data privacy concerns. A similar data privacy concern appeared recently in the news about Sidewalk Labs, which is a smart-city company owned by Google. With the unprecedented growth of the Internet in all aspects of life, the advent of phenomenons such as the Internet of Things (IoT) and big data, it is anticipated that more data privacy challenges will be raised in coming years.

To address the data privacy concerns, the root causes of the problem should first be understood. In the case of digital data in the information age, the problem

starts when appropriate data privacy measures are not utilized in the data and computational infrastructures. In particular, the data owners store their data on the data centers or on the cloud owned by third parties. This data will then be used by the owners of the data centers for different purposes, e.g. advertisement, commercial and data analytics goals. It can be said that this is one of the primary sources of the problem.

Addressing data securiy and privacy concerns has been the focus of attention by many researchers from decades ago [30, 37, 38]. In [30], the pioneers of cryptography discussed how privacy-preserving computation using homomorphisms can be achieved. In [37] and [38], on the other hand, the idea of secure two-party computation was initiated. In particular, in [37], the *Yao's Millionaires problem* was introduced and in [38] the *Yao's garbled circuit* technique was developed to solve the Yao's Millionaires problem. The above seminal ideas led to the general idea of *secure multiparty computation* (secure MPC). Secure MPC is a cryptographic technique which enables a group of parties to evaluate a function based on the private data that each party provides, for instance in [26]. This technique can help us address data security and privacy issues to a good extent.

1.1 Our Contribution

There has been significant amount of theoretical and practical advancements in secure multiparty computation. Nowadays, there are various implementations and libraries of secure MPC frameworks for real-world applications. Each library is based on different theoretical approaches and works in certain settings. This is mainly because of the strengths and shortcomings of different MPC approaches. In spite of the amount of conducted research, the existing literature rarely pointed out the capabilities and incapabilities of different MPC solutions. For instance, in [23] the authors claimed that many existing MPC frameworks fail to work properly in practice, because of crashing or generating incorrect circuits [23]. A technical review comparing different approaches and highlighting both strengths and weaknesses of the MPC solutions is missing in the literature.

In this paper, we provide a technical review of secure MPC tools that can be used in the IoT and big data analytics. The contribution of this comparative and technical review is multi-fold. On the theoretical side, this paper highlights the strengths and weaknesses of different cryptographic techniques that are commonly used in MPC tools. It also delves into the main arithmetic operations that are carried out at the core of MPC protocols. On the practical side, this paper reviews the state-of-the-art MPC tools. We provide tables for summarizing and comparing the existing MPC tools, their security adversarial models, and the application domains in which each tool can be utilized. The paper also highlights potential approaches for the future secure MPC frameworks.

We would like to emphasize that in this paper we provide a very technical review of the secure MPC tools and libraries with an emphasis on the cryptographic primitives and mathematical/arithmetic operations that are performed at the core of secure MPC protocols. There are other comprehensive studies, including the recent ones [2] and [33], that have studied secure MPC tools from

different perspectives. In [33], the authors studied the applications of privacy-preserving computation in fog computing. While in [2], a thorough analysis of secure MPC solutions and its relevance to other privacy-preserving computation areas, e.g. differential privacy, has been provided. In this technical review, our goal is to instill in the reader an enhanced understanding of different approaches in applying secure MPC solutions to the IoT and big data analytics.

2 Different approaches for Secure Computation

There are different approaches to implement a secure multiparty computation scheme. These techniques are based on three cryptographic primitives, i.e., secret sharing [32], homomorphic encryption [25, 30] and Yao's garbled circuits [38].

2.1 Secure Computation based on Secret Sharing

Secret sharing is one of the dominant approaches used in secure multiparty computation. In this approach, the participating parties use a secret sharing scheme, e.g., Shamir's scheme [32], to share their secrets (private data). In order to emulate a secure MPC protocol, the parties then perform computations on the shares of their data, rather than directly on their data. Since the shares of the private data are random values, no information about that data is revealed.

An advantage of secure computation protocols based on secret sharing is that such protocols can provide information-theoretical security given that the underlying scheme is information-theoretically secure. Another advantage of MPC based on secret sharing is that there is no need for any encryption/decryption key. However, secret-sharing-based MPC protocols require significant amount of communication among the participating parties. In fact, privacy is achieved by distributing the computations among the parties.

2.2 Secure Computation based on Homomorphic Encryption

Another commonly-used approach in secure multiparty computation protocols is homomorphic encryption. In this case, the parties utilize a homomorphic encryption scheme, e.g., the Paillier scheme [25], to encrypt their data. The parties then perform computations on the encrypted form of data. Homomorphic encryption has attracted significant attention in the last decade. In particular, by the appearance of fully homomorphic encryption (FHE) schemes [15], this research area has shown to be more promising.

Most of the cryptographic schemes are based on the difficulty of some computational problems, e.g., integer factorization or discrete logarithm. This can be considered as one of the drawbacks of homomorphic-encryption-based MPC protocols. This is due to the fact that if the underlying difficult problem is solved (e.g., by utilizing quantum computers), the encryption scheme would not be secure anymore. In addition, homomorphic-encryption-based MPC protocols are computationally intensive and supporting multi-key encryption is a challenging task in such schemes [1].

2.3 Secure Computation based on Yao's Garbled Circuits

Yao's garbled circuits [38] is another dominant approach for secure two-party computation. A garbled circuit is an encrypted form of a function, which is supposed to be evaluated securely between two parties. More precisely, in this approach, one party encrypts the bits of their input and the intermediate state of the computation. This party then converts the computation into a circuit of binary gates, each represented as a garbled truth table. The other party, a.k.a., the evaluator, receives the circuit and the encrypted input bits. The evaluator then produces the encrypted output by evaluating each gate at the encrypted bits of the input and combining the results.

Yao's garbled circuit approach is the most efficient method for securely evaluating boolean circuits [20]. This approach does not require any communication between the parties during the evaluation. However, the intermediate state in the garbled circuits is far larger than the input data. This makes garbled circuits impractical for processing large data. Moreover, the garbled circuit approach provides computational security.

3 Building Blocks for Secure Computation

At the core of the three common approaches in secure computation, the main arithmetic operations, i.e., addition, multiplication, subtraction, division and comparison, are performed. These arithmetic operations, in fact, form the building blocks of secure computation.

3.1 Secure Comparison

Secure comparison is an important building block in secure computation [31]. The problem of secure comparison was initially introduced in [37], as the Yao's Millionaires problem. This problem is a well-studied, but challenging problem. Thus far, different solutions have been proposed to this problem. The proposed solutions are mostly based on homomorphic encryption techniques, secret sharing schemes and Yao's garbled circuits.

The current solutions to the secure comparison problem are very expensive, mostly in terms of the communication complexity. An inefficient secure comparison protocol can make a secure multiparty computation protocol even more inefficient. This is due to the fact that secure comparison may be used numerous times in a MPC protocol. For instance, secure comparison is frequently used in the secure argmax operation, which is another common operation in many privacy-preserving data mining algorithms [6]. Thus, efficient and practical solutions to the secure comparison problem result in improving secure computation protocols. There are different approaches for improving a secure comparison protocol. For example, reducing the number of interactions among the participating parties is a potential optimization technique.

3.2 Other Building Blocks for Secure Computation

For performing secure computation, the four main arithmetic operations need to be implemented in a secure fashion. These operations can be implemented securely using secret sharing schemes, homomorphic encryption techniques and Yao's garbled circuits. For instance, the Paillier homomorphic encryption scheme [25] allows us to calculate the addition of two encrypted values by multiplying their corresponding ciphertexts and without decrypting them. In the case of secret sharing schemes, two or more parties can calculate the addition of their secret values by adding the shares of the secret values locally and then conducting a Lagrange interpolation on their updated shares.

Depending on the application domain, the secure implementation of other operations may also be needed. For instance, in privacy-preserving data mining and machine learning, the secure version of three operations is needed. These operations include secure comparison, secure inner product of two vectors, and secure argmax [6]. In some cases, the secure version of natural logarithm, i.e. the *ln()* function, the sign function, the sigmoid function is also required [9].

4 Security & Privacy Challenges in IoT and Big Data

Data security and privacy have been critical challenges both in the Internet of Things (IoT) and big data domains [4, 34, 40]. It is important to scrutinize major security and privacy issues in these domains. A good understanding of such issues helps us provide concrete solutions to the problems. The Cloud Security Alliance [4] has included secure computations and cryptographic solutions among the top ten challenges to big data security & privacy. Moreover, data privacy has been indentified as one of the major security concerns [4].

Addressing data security and privacy issues is a challenging task. Three primary challenges of using secure multiparty computation frameworks in the big data domain are as follows [36]:

1. MPC is not integrated well with current data processing and data analytic workflows
2. Significant expert knowledge is needed for implementing and running data analytics in the MPC frameworks
3. The MPC frameworks do not scale well for large data sets, because large-data processing systems do not support efficient parallel processing yet

In addition to the aforementioned challenges, there are still some limitations with secure multiparty computation schemes that preclude using them in the IoT and big data domains. First of all, providing a general-purpose efficient MPC framework for various applications in different domains has shown to be very difficult. There has been tremendous amount of research on secure computation for different applications, including privacy-preserving data mining, sealed-bid auctions, privacy-preserving face recognition, and private information retrieval, to name a few. Secure multiparty computation protocols have been used in

different application domains with specific settings and assumptions depending on the suitability and efficiency criteria. Combining these solutions to have an integrated framework is quite challenging. The challenges that IoT and big data analytics bring will be added and will make the scenario even more complicated [19]. Nonetheless, a careful combination of different solutions might be a plausible approach in the near future.

5 Tools for Privacy-Preserving Big Data Analytics

In this section, we compare the state-of-the-art secure multiparty computation tools (including libraries, implementations and frameworks). These tools can be used for secure computation in the IoT and big data domains [2, 33].

Fairplay [22] is a secure function evaluation (SFE) tool that allows two parties to perform a joint computation without any trusted third party. This tool is based on Yao's garbled circuits and provides a high-level function description language called SFDL. The Fiarplay compiler compiles SFDL programs into a boolean circuit and evaluates the circuit using its runtime environment.

FairplayMP [3] is an extension of Fairplay [22] for multiple parties. This tool is based on Yao's garbled circuits and secret sharing schemes. FairplayMP uses an emulated trusted third party. The emulated trusted third party receives the inputs from the parties, does the desired computations and privately informs the parties of their outputs.

Sharemind [5] is a secure multiparty computation framework consisting of three parties. It is one of the most developed and efficient MPC tools and supports 32-bit integer arithmetic. However, it uses a non-standard secret sharing technique and does not extend to more than three parties [41].

VIFF [10] is a compiler for secure multiparty computation based on standard secret sharing schemes. It uses parallelization and multi-threading to provide faster computations. This framework supports computations consisting of basic primitives, e.g. addition and multiplication, on secret-shared values.

SEPIA [8] is a Java library based on linear secret sharing schemes. It separates the parties into computational parties and the parties who provide inputs and obtain outputs. SEPIA is used for secure distributed computation on network data, e.g., for privacy-preserving network intrusion detection.

TASTY [17] is a tool (with a compiler) for two-party secure computation (2PC) based on Yao's garbled circuits and homomorphic encryption. This tool can be used for describing, generating, executing, benchmarking and comparing secure 2PC protocols. It allows a user to provide a description of the computations to be performed and transforms the description into a 2PC protocol.

SPDZ [11] is a secure multiparty computation protocol based on secret sharing and homomorphic encryption. SPDZ consists of an offline (preprocessing) phase and an online phase. In the offline phase, the required shared random data is generated and in the online phase, the actual secure computation is carried out.

SCAPI [13] is an open-source library for developing MPC frameworks and secure computation implementations. It comes with two instantiations of the Yao's garbled circuits. One instantiation is secure against active adversaries and the other is secure against passive adversaries. SCAPI is implemented in Java and uses the JNI framework for calling native codes, to make the library efficient.

Wysteria [27] is a high-level programming language for writing MPC programs. It supports mixed-mode programs consisting of private computations with multiparty computations. Wysteria compiles the MPC programs to circuits and then executes the circuits by its underlying MPC engine.

Obliv-C [39] is a language for secure computation programming based on the garbled circuits. It is an extension of the C programming language that provides data-oblivious programming constructs. The Obliv-C compiler, implemented as a modified version of CIL, transforms Obliv-C codes to plain C codes.

Enigma [42] is a decentralized computation framework which combines MPC and Blockchain technology to provide guaranteed privacy. It allows different parties to jointly store and perform computations on their data without exposing the privacy of the data. Enigma also removes the need for trusted third parties.

Frigate [23] is a validated compiler and fast circuit interpreter for secure computation. It introduces a C-style language for secure function evaluation based on garbled circuits. Frigate has been developed with an emphasis on the principles of compiler design. It addresses the limitations of many previous MPC frameworks and produces correct and functioning circuits [23].

Chameleon [29] is a hybrid framework for privacy-preserving machine learning. This framework is based on the ABY framwork [12], which implements a combination of secret sharing, garbled circuits and the GMW protocol [16]. Chameleon has an offline and an online phase and most of the computation is performed in the offline phase. It uses a semi-honest third party (STP) in the offline phase, for generating the required correlated random values.

Wys* [28] is a domain-specific language (DSL) for writing mixed-mode secure MPC programs. It is based on the the idea of Wysteria [27] and embedded/hosted in F* programming language. For running a MPC program in Wys*, the program is first compiled using the F* compiler. Then each party runs the compiled codes using the Wys* interpreter. The result, which is a boolean circuit, is evaluated using the GMW protocol [16] on the parties' secret shares.

Conclave [35] is a query compiler that makes secure computation on big data efficient. Conclave generates codes for cleartext processing in Python and Spark and codes for secure computation using Sharemind [5] and Obliv-C [39]. The idea behind Conclave is to minimize the computations under MPC as much as possible. Conclave can support only two or three parties and withstands a passive semi-honest adversary.

We summarized the reviewed secure MPC tools in Table 1. The table illustrates the main details and characteristics of the tools. Note that, due to the space constraints, we used some abbreviations in Table 1. The meaning of the abbreviations is provided in Table 2.

The first column of Table 1 shows the name of the MPC tools, the year in which each tool was developed, and the reference related to each tool. The second column determines the number of parties that each tool supports. The third column specifies the cryptographic primitives that have been used in the development of each tool. The fourth column defines the type of security, i.e. computational or information-theoretical, that each tool provides. The fifth column shows whether each tool uses some trusted third party (TTP) or such a party is simulated in the tool. The idea of doing secure multiparty computation without relying on any trusted third party is an interesting one. However, realizing such a computational model seems to be a challenging task; as the fifth column of Table 1 shows, the majority of the listed tools either need trusted third parties or simulate them. The last column of the table shows the programming languages that were used for the development of each tool.

We also provided a table that illustrates the adversarial model for each MPC tool; see Table 3. The table specifies the number of corrupted parties that each tool can tolerate. Note that in secure multiparty computation, the participating parties might be corrupted by some adversaries. The parties may also collude with each other. Therefore, it is important to consider such scenarios in the implementation. Finally, Table 4 shows some applications for each MPC tool.

Tool/Library	**Parties**	**Based on**	**Security**	**TTP**	**Prog. Lang.**
Fairplay 2004 [22]	2	GC	Computational	Yes	SFDL (Java)
FairplayMP 2008 [3]	≥ 3	GC and SS	Computational	Em. TTP	SFDL (Java)
Sharemind 2008 [5]	3	Additive SS	Info. Theortic	Yes	SecreC (C++)
VIFF 2009 [10]	≥ 3	SS	Info. Theortic	No	Python
VIFF 2009 [10]	2	Paillier HE scheme	Computational	No	Python
SEPIA 2009 [8]	≥ 3	Shamir's SS	Computational	Sim. TTP	Java
TASTY 2010 [17]	2	HE and GC	Computational	No	Python
SPDZ 2012 [11]	≥ 2	SS and HE	Computational	Yes	C++/Python
SCAPI 2012 [13]	≥ 2	GC	Computational	No	Java
Wysteria 2014 [27]	≥ 2	GMW protocol	Info. Theortic	Sim. TTP	OCaml
Obliv-C 2015 [39]	2	GC	Computational	No	C
Enigma 2015 [42]	≥ 2	VSS and Blockchain	Info. Theortic	No	WebAssembly
Frigate 2016 [23]	2	GC	Computational	No	C++
Chameleon 2018 [29]	2	SS, GMW, GC	Computational	STP	C++
Wys* 2019 [28]	≥ 2	[27]	Info. Theortic	Sim. TTP	F*
Conclave 2019 [35]	2 or 3	[5] and [39]	Computational	Yes	Python/Spark

Table 1. Secure MPC Tools for Big Data Computation (based on [33])

Notation	**Meaning**
HE	Homomorphic Encryption
GC	Yao's Garbled Circuits
GMW	the Goldreich, Micali, and Wigderson (GMW) protocol [16]
SFDL	Secure Function Definition Language
SS	Secret Sharing
VSS	Verifiable Secret Sharing
STP	Semi-honest Third Party
TTP	Trusted Third Party
Em. TTP	Emulated Trusted Third Party
Sim. TTP	Simulated Trusted Third Party

Table 2. Abbreviations used in Table 1

Tool/Library	Secure against
Fairplay 2004 [22]	not mentioned
FairplayMP 2008 [3]	a collection of $\lfloor \frac{n}{2} \rfloor$ corrupt computation players, as long as they operate in a semi-honest way
Sharemind 2008 [5]	a passive adversary able to corrupt at most one party
VIFF 2009 [10]	not mentioned
SEPIA 2009 [8]	$t < \frac{m}{2}$ colluding privacy peers. Note that the systems has n input peers and m privacy peers
TASTY 2010 [17]	not mentioned
SPDZ 2012 [11]	an active adversary capable of corrupting up to $(n-1)$ parties
SCAPI 2012 [13]	both active and passive adversaries
Wysteria 2014 [27]	a semi-honest adversary capable of corrupting up to $(n-1)$ parties
Obliv-C 2015 [39]	semi-honest adversaries
Enigma 2015 [42]	not mentioned
Frigate 2016 [23]	semi-honest model
Chameleon 2018 [29]	semi-honest (honest-but-curious) model
Wys* 2019 [28]	semi-honest (honest-but-curious) model
Conclave 2019 [35]	a passive semi-honest adversary

Table 3. Table of Adversarial Model

Tool/Library	Applications
Fairplay 2004 [22]	secure two-party computation
FairplayMP 2008 [3]	secure multiparty computation
Sharemind 2008 [5]	tax fraud detection system
VIFF 2009 [10]	sugar beet auction, decision tree learning, privacy-preserving verifiable computation
SEPIA 2009 [8]	private information aggregation, network security and monitoring
TASTY 2010 [17]	set intersection, face recognition
SPDZ 2012 [11]	oblivious RAM schemes and oblivious data structures for MPC
SCAPI 2012 [13]	privacy-preserving impersenation detection systems and fair exchange protocols
Wysteria 2014 [27]	DStress (a framework for privacy-preserving and distributed graph analytics)
Obliv-C 2015 [39]	secure computation and data-oblivious computation
Enigma 2015 [42]	decentralized computation, IoT, crypto bank, blind e-voting, n-factor authentication
Chameleon 2018 [29]	privacy-preserving machine learning, e.g. SVM and deep learning
Wys* 2019 [28]	joint median, card dealing, private set intersection (PSI)
Conclave 2019 [35]	secure MPC on big data, e.g. credit card regulation and market concentration

Table 4. Table of Applications

6 Technical Discussion and Future Works

6.1 Technical Discussion

There are three common approaches for implementing secure MPC protocols. These approaches include: secret sharing schemes, Yao's garbled circuits, and homomorphic encryption techniques. The approaches that work based on secret sharing and homomorphic encryption schemes usually use the so-called arithmetic gates, i.e. *Addition* and *Multiplication* gates. While, the approaches that work based on Yao's garbled circuits usually encrypt the inputs and garble the circuit of the function which is supposed to be securely computed. Although the three MPC approaches determine the overall schema for secure computation, sometimes it is preffered to securely implement certain functionalities. For

instance, in the case of privacy-preserving data mining and machine learning, the three commonly-used operations [6], include secure comparison, secure inner product of two vectors, and secure argmax. Other common functions that may need to be implemented securely include the sigmoid function, the sign function and the floor function [9] and [7].

Secure comparison is an arithmetic operation which commonly appears in almost any secure computation protocol. Secure comparison is in fact the Yao's Millionaires problem [37], which is a well-studied problem. However, most of the secure comparison solutions are expensive in terms of the communication complexity, i.e., interaction among the parties. According to [31], secure comparison protocols based on additive homomorphic encryption schemes require significant amount of interaction among the parties. This is because additive homomorphic encryption schemes allow linear operations (i.e., addition or multiplication by a constant) on the encrypted values; whereas comparison is a non-linear arithmetic operation. The inefficiency of secure comparison protocols makes the secure argmax operation inefficient as well; and thus, the secure multiparty computation protocols. Therefore, providing an efficient solution to secure comparison can bring in a significant improvement for secure MPC protocols.

An advantage of secure MPC tools based on Yao's garbled circuits is that they are fast. However, such tools do not provide information-theoretical security and they are mostly used for secure two-party computations. Whereas, MPC tools based on secret sharing schemes provide information-theoretical security given that the underlying scheme is information-theoretically secure. MPC solutions based on secret sharing schemes do not need any cryptographic key. However, such tools require significant amounts of interactions among the parties. MPC tools based on homomorphic encryption techniques provide computational security. Early homomorphic encryption schemes, e.g., Paillier homomorphic encryption scheme [25], cannot support both addition and multiplication operations, which are required for secure multiparty computation. This reduces their applicability in secure MPC tools. Recent homomorphic encryption techniques, e.g., fully homomorphic encryption (FHE) [15], can support a limited number of addition and multiplication gates. In addition, an overlooked drawback of the FHE schemes is that they rarely support multi-key encryption [1].

MPC protocols based on secret sharing and those based on homomorphic encryption schemes work based on arithmetic gates, i.e. *Addition* and *Multiplication* gates. The multiplication gate in such protocols has shown to make the computations inefficient. For instance, for doing a multiplication in a MPC protocol based on Shamir's secret sharing, the participating parties must regularly perform a process called degree reduction. Similarly, in MPC protocols based on fully homomorphic encryption (FHE) schemes, the parties must regularly carry out a noise reduction process (i.e. bootstrapping) in order for the FHE schemes to work properly. In both cases, the degree reduction and the noise reduction processes deteriorate the performance of MPC protocols drastically.

6.2 Future Works

There are different interesting avenues for further research. One line of research is to evaluate and test the existing MPC solutions in different application domains, with the purpose of improving such solutions. For instance, one can perform experimental research using the recent MPC prototypes, e.g., Enigma [42], which is decentralized thanks to the Blockchain technology. Another direction is to focus on the main arithmetic operations which are run at the core of MPC protocols. For instance, providing efficient solutions for secure comparison can improve the efficiency of the MPC solutions. Improving the multiplication gate of MPC solutions based on homomorphic encryption or secret sharing schemes can also result in more efficient and practical MPC solutions.

Another very interesting line of research is to integrate social mechanisms, e.g., trust and reputation, in secure MPC protocols. Utilizing social mechanisms alongside secure MPC protocols can help us achieve more secure and trustworthy data and computation frameworks [18, 43]. This is because trust and reputation are considered as soft security measures that compliment hard security measures, e.g. cryptography and secure MPC protocols. In particular, secure MPC solutions combined with trust and reputation machanisms can be helpful in trustworthy machine learning. It is worth mentioning that trustworthiness in data analytics and machine learning techniques is becoming more important as we rely more on such techniques [21, 24]. Integrating secure MPC protocols into emerging decentralized computation technologies, e.g., the Blockchain technology, can also be another potential line of research [42].

One may also do further research on more efficient secure solutions to the commonly-used operations/functions in data mining and machine learning techniques. However, for achieving such solutions it may be needed to accept a trade-off between approximation and efficiency. In [9], for instance, the authors faced some challenges for implementing logistic regression over encrypted data. According to [9], the homomorphic implementation of the sign function, which is closely related to the comparison operator, is very difficult. Implementing the sigmoid function using homomorphic encryption seems also to be very difficult. Note that the sigmoid function is commonly used in neural networks' activation functions and in logistic regression models. Even more challenging seems to be the floor function [9]. The authors [9] dealt with these challenges using polynomial approximation, e.g., Taylor polynomials and minimax approximation [14]. An interesting line of research is to see how such approximation methods will perform for the functions that are commonly used in the secure MPC protocols. For instance, one may further study the approximation solutions for the secure comparison and secure argmax operations.

7 Conclusion

Data security and privacy have been crucial issues in recent years. It can be said that these issues will become even more crucial as we are going well into the Internet of Things (IoT) and big data eras. One of the main causes of data

privacy violation is due to not utilizing appropriate data privacy measures in the data and computational infrastructures. Secure multiparty computation (secure MPC) is a powerful cryptographic tool that can help us address data security and privacy issues. In this paper, we provided a technical review to the cryptographic techniques commonly used in MPC protocols. We delved into the arithmetic operations that are run at the core of secure MPC protocols. In addition, we highlighted the strengths and weaknesses of different approaches used in secure MPC, and the challenges we face for designing practical MPC solutions. We also compared the state-of-the-art MPC tools that can be used for addressing security and privacy issues in the IoT and big data domains.

Considering all aspects and challenges of secure computation, solutions based on secret sharing schemes integrated into decentralized computation frameworks seem to be more promising for the future. Such solutions are decentralized, provide information-theoretical security and do not need any cryptographic key. Enigma [42] might be considered as a sample proof-of-work and a prototype for potential practical solutions. In addition, integrating social mechanisms, such as trust and reputation, into secure multiparty computation protocols provides more reliable and trustworthy data and computation frameworks. Our technical review and comparative study of different secure MPC approaches and developed MPC tools will have significant contributions to applying secure MPC solutions to the IoT and big data domains.

References

1. Acar, A., Aksu, H., Uluagac, A.S., Conti, M.: A survey on homomorphic encryption schemes: Theory and implementation. ACM Computing Surveys (CSUR) **51**(4), 79 (2018)
2. ALX, P.S.N., ALX, N.V., AU, P.F., AU, C.O., AU, P.S., AU, M.S., PHI, M.V., TUE, N.B., TUE, B.S.: D1. 1 state of the art analysis of mpc techniques and frameworks
3. Ben-David, A., Nisan, N., Pinkas, B.: Fairplaymp: a system for secure multi-party computation. In: Proceedings of the 15th ACM conference on Computer and communications security. pp. 257–266. ACM (2008)
4. BigDataWorkingGroup: Expanded top ten big data security and privacy challenges. `https://downloads.cloudsecurityalliance.org/initiatives/bdwg/Expanded_Top_Ten_Big_Data_Security_and_Privacy_Challenges.pdf` (2013)
5. Bogdanov, D., Laur, S., Willemson, J.: Sharemind: A framework for fast privacy-preserving computations. In: European Symposium on Research in Computer Security. pp. 192–206. Springer (2008)
6. Bost, R., Popa, R.A., Tu, S., Goldwasser, S.: Machine learning classification over encrypted data. In: NDSS. vol. 4324, p. 4325 (2015)
7. Bourse, F., Minelli, M., Minihold, M., Paillier, P.: Fast homomorphic evaluation of deep discretized neural networks. In: Annual International Cryptology Conference. pp. 483–512. Springer (2018)
8. Burkhart, M., Strasser, M., Many, D., Dimitropoulos, X.: Sepia: Privacy-preserving aggregation of multi-domain network events and statistics. Network **1**(101101) (2010)

9. Chen, H., Gilad-Bachrach, R., Han, K., Huang, Z., Jalali, A., Laine, K., Lauter, K.: Logistic regression over encrypted data from fully homomorphic encryption. BMC medical genomics **11**(4), 81 (2018)
10. Damgård, I., Geisler, M., Krøigaard, M., Nielsen, J.B.: Asynchronous multiparty computation: Theory and implementation. In: International Workshop on Public Key Cryptography. pp. 160–179. Springer (2009)
11. Damgård, I., Pastro, V., Smart, N., Zakarias, S.: Multiparty computation from somewhat homomorphic encryption. In: Annual Cryptology Conference. pp. 643–662. Springer (2012)
12. Demmler, D., Schneider, T., Zohner, M.: Aby-a framework for efficient mixed-protocol secure two-party computation. In: NDSS (2015)
13. Ejgenberg, Y., Farbstein, M., Levy, M., Lindell, Y.: Scapi: The secure computation application programming interface. IACR Cryptology EPrint Archive **2012**, 629 (2012)
14. Fraser, W.: A survey of methods of computing minimax and near-minimax polynomial approximations for functions of a single independent variable. Journal of the ACM (JACM) **12**(3), 295–314 (1965)
15. Gentry, C., et al.: Fully homomorphic encryption using ideal lattices. In: Stoc. vol. 9, pp. 169–178 (2009)
16. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game. In: Proceedings of the nineteenth annual ACM symposium on Theory of computing. pp. 218–229. ACM (1987)
17. Henecka, W., Sadeghi, A.R., Schneider, T., Wehrenberg, I., et al.: Tasty: tool for automating secure two-party computations. In: Proceedings of the 17th ACM conference on Computer and communications security. pp. 451–462. ACM (2010)
18. Jøsang, A., Ismail, R., Boyd, C.: A survey of trust and reputation systems for online service provision. Decision support systems **43**(2), 618–644 (2007)
19. Kaisler, S., Armour, F., Espinosa, J.A., Money, W.: Big data: Issues and challenges moving forward. In: 2013 46th Hawaii International Conference on System Sciences. pp. 995–1004. IEEE (2013)
20. Kolesnikov, V., Sadeghi, A.R., Schneider, T.: Improved garbled circuit building blocks and applications to auctions and computing minima. In: International Conference on Cryptology and Network Security. pp. 1–20. Springer (2009)
21. LaValle, S., Lesser, E., Shockley, R., Hopkins, M.S., Kruschwitz, N.: Big data, analytics and the path from insights to value. MIT sloan management review **52**(2), 21 (2011)
22. Malkhi, D., Nisan, N., Pinkas, B., Sella, Y., et al.: Fairplay-secure two-party computation system. In: USENIX Security Symposium. vol. 4, p. 9. San Diego, CA, USA (2004)
23. Mood, B., Gupta, D., Carter, H., Butler, K., Traynor, P.: Frigate: A validated, extensible, and efficient compiler and interpreter for secure computation. In: 2016 IEEE European Symposium on Security and Privacy (EuroS&P). pp. 112–127. IEEE (2016)
24. Najafabadi, M.M., Villanustre, F., Khoshgoftaar, T.M., Seliya, N., Wald, R., Muharemagic, E.: Deep learning applications and challenges in big data analytics. Journal of Big Data **2**(1), 1 (2015)
25. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: International Conference on the Theory and Applications of Cryptographic Techniques. pp. 223–238. Springer (1999)

26. Raeini, M.G., Nojoumian, M.: Secure error correction using multiparty computation. In: 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC). pp. 468–473. IEEE (2018)
27. Rastogi, A., Hammer, M.A., Hicks, M.: Wysteria: A programming language for generic, mixed-mode multiparty computations. In: 2014 IEEE Symposium on Security and Privacy. pp. 655–670. IEEE (2014)
28. Rastogi, A., Swamy, N., Hicks, M.: Wys*: : A dsl for verified secure multi-party computations. In: International Conference on Principles of Security and Trust. pp. 99–122. Springer (2019)
29. Riazi, M.S., Weinert, C., Tkachenko, O., Songhori, E.M., Schneider, T., Koushanfar, F.: Chameleon: A hybrid secure computation framework for machine learning applications. In: Proceedings of the 2018 on Asia Conference on Computer and Communications Security. pp. 707–721. ACM (2018)
30. Rivest, R.L., Adleman, L., Dertouzos, M.L., et al.: On data banks and privacy homomorphisms. Foundations of secure computation **4**(11), 169–180 (1978)
31. Schneider, M., Schneider, T.: Notes on non-interactive secure comparison in image feature extraction in the encrypted domain with privacy-preserving sift. In: Proceedings of the 2nd ACM workshop on Information hiding and multimedia security. pp. 135–140. ACM (2014)
32. Shamir, A.: How to share a secret. Communications of the ACM **22**(11), 612–613 (1979)
33. Sousa, P.R., Antunes, L., Martins, R.: The present and future of privacy-preserving computation in fog computing. In: Fog Computing in the Internet of Things, pp. 51–69. Springer (2018)
34. Tonyali, S., Akkaya, K., Saputro, N., Uluagac, A.S., Nojoumian, M.: Privacy-preserving protocols for secure and reliable data aggregation in iot-enabled smart metering systems. Future Generation Computer Systems **78**, 547–557 (2018)
35. Volgushev, N., Schwarzkopf, M., Getchell, B., Varia, M., Lapets, A., Bestavros, A.: Conclave: secure multi-party computation on big data. In: Proceedings of the Fourteenth EuroSys Conference 2019. p. 3. ACM (2019)
36. Volgushev, N., Schwarzkopf, M., Lapets, A., Varia, M., Bestavros, A.: integrating mpc in big data workflows. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. pp. 1844–1846. ACM (2016)
37. Yao, A.C.C.: Protocols for secure computations. In: FOCS. vol. 82, pp. 160–164 (1982)
38. Yao, A.C.C.: How to generate and exchange secrets. In: 27th Annual Symposium on Foundations of Computer Science (sfcs 1986). pp. 162–167. IEEE (1986)
39. Zahur, S., Evans, D.: Obliv-c: A language for extensible data-oblivious computation. IACR Cryptology ePrint Archive (2015)
40. Zarpelao, B.B., Miani, R.S., Kawakani, C.T., de Alvarenga, S.C.: A survey of intrusion detection in internet of things. Journal of Network and Computer Applications **84**, 25–37 (2017)
41. Zhang, Y., Steele, A., Blanton, M.: Picco: a general-purpose compiler for private distributed computation. In: Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security. pp. 813–826. ACM (2013)
42. Zyskind, G., Nathan, O., Pentland, A.: Enigma: Decentralized computation platform with guaranteed privacy. arXiv preprint arXiv:1506.03471 (2015)
43. Zyskind, G., Nathan, O., et al.: Decentralizing privacy: Using blockchain to protect personal data. In: Security and Privacy Workshops (SPW), 2015 IEEE. pp. 180–184. IEEE (2015)