Secure Multi-Party Computation for Analytics Deployed as a Lightweight Web Application

Andrei Lapets, Nikolaj Volgushev, Azer Bestavros, Frederick Jansen, Mayank Varia
Email: {lapets, nikolaj, best, fjansen, varia}@bu.edu
CS Dept., Boston University
111 Cummington Mall
Boston, MA USA 02215

***Abstract*—We describe the definition, design, implementation, and deployment of a secure multi-party computation protocol and web application. The protocol and application allow groups of cooperating parties with minimal expertise and no specialized resources to compute basic statistical analytics on their collective data sets without revealing the contributions of individual participants. The application was developed specifically to support a Boston Women's Workforce Council (BWWC) study of wage disparities within employer organizations in the Greater Boston Area. The application has been deployed successfully to support two data collection sessions (in 2015 and in 2016) to obtain data pertaining to compensation levels across genders and demographics. Our experience provides insights into the particular security and usability requirements (and tradeoffs) a successful "MPC-as-a-service" platform design and implementation must negotiate.**

I. Introduction

Companies, educational institutions, government agencies, and other modern organizations have been collecting and analyzing data pertaining to their internal operations for some time and with great effect, such as in evaluating performance or improving efficiency. While each organization's own data has internal value, combining data from multiple organizations and analyzing it as a single corpus is likely to provide even more value to the organizations themselves, to policymakers, or to society at large.

Unfortunately, each organization's internal data is often proprietary and confidential, and its release may be potentially deleterious to the organization's interests. Organizations may have the option of releasing sensitive data selectively to specific agents entrusted with its analysis, but this is often costly, requires that the organizations strongly trust the agent, and presents a security risk if the data is improperly protected at rest.

Secure multi-party computation (MPC) resolves this tension: aggregate data may be computed and released while preserving the confidentiality of each organization's internal data. Theoretical constructs for MPC have been known for decades [1]–[4], and recent efforts [5]–[9] are finally bringing us closer to a point at which these techniques will be available to end-users (i.e., organizations interested in collectively analyzing their sensitive data).

In this paper, we describe the design of a relatively modest MPC protocol along with (more importantly) the implementation and deployment of a web-based service to compute aggregate metrics without requiring the organizations to trust that service with any sensitive data. We successfully deployed this service twice to analyze compensation data (broken down by gender and demographics) from a collection of 40 to 70 employer organizations.

The definition of the protocol and the design and implementation of the software were influenced heavily by the practical restrictions and challenges of the particular target application. In particular, human factor constraints such as comprehension and usability, presented difficulties that are typically ignored in the design of secure protocols and by existing MPC frameworks, which are invariably intended for users with at least some technical expertise and computing resources.

As a result, we advocate for the design of secure MPC as a service, with a powerful service provider connected to data holders and data analysts in a star topology. The service provider acts as a "smart router": without learning any sensitive data, its computational power and network accessibility enables the other participants to calculate an analytic using nothing more than a web browser.

II. Scenario and Requirements

In this section, we highlight the scenario that motivated our work, describe the parties involved in secure MPC, and survey the security and usability guarantees that we require of an implementation of secure MPC. We remark that these requirements overlap: usability of security assertions is important, as otherwise no one understands what is being guaranteed and what is not.

A. Application Scenario

The Boston Women's Workforce Council (BWWC) initiated a study of gender and ethnicity wage gaps among employers within the Greater Boston Area. As part of this study, compensation data had to be collected from privately held companies in order to calculate an aggregate statistic (sum) over the data points. Each participating company served as a contributor that submitted cumulative employee earnings aggregated by gender and job category.

BWWC takes on the role of the dedicated data analyzer. They may view the employee earnings totals aggregated across *all* companies; however, the individual company aggregates must remain private and must not be revealed to any single party. The Hariri Institute at Boston University serves as an (untrusted) service provider, supplying the computational architecture and personnel resources to facilitate the private aggregation.

B. Parties in Secure MPC and their Trust Relationships

Generalizing from the pay equity scenario, we consider three types of roles in secure multi-party computation.

- Several *contributors* who contribute private data for the calculation of the analytic. The number of contributors is unbounded and may not be known in advance.
- An automated, publicly-accessible *service provider* that connects all other participants without requiring them to maintain servers (or even to be online simultaneously), and that partially calculates the analytic.
- One or more *analyzers* who receive the output of the analytic and may also help in computing it.

Both the contributors and analyzers must place some trust in each other. Analyzers trust that the contributors submit valid data, while contributors both trust the analyzers to protect the aggregate output and trust that each analyzer will not collude with anybody else. If more than one analyzer is supported, then the contributors' trust is *federated* rather than added: the contributors need only trust that one of the analyzers is honest.

Critically, nobody needs to entrust the service provider with any of their data. They only need to believe that the service provider is incentivized to perform the calculation on their behalf (i.e., denial of service is out of scope).

C. Security and Privacy Requirements

We rely on multi-party computation with passive (a.k.a. semi-honest) security and without collusion. This security notion [10] essentially states that parties cannot learn any data other than what may be inferred from the aggregate analytic as long as they all adhere to the protocol.

Passive security suffices in our scenario because the service provider and analyzer lack any clear incentive to falsify the results of the aggregation or to learn private input data. On the contrary, completing the study successfully is directly beneficial to BWWC (as the initiator of the study) as well as to the Hariri Institute (as a research institute reliant upon a reputation of integrity). Additionally, obtaining any of the contributors' private data creates a liability risk for the service provider and analyzer. The semi-honest model is very natural in this case as it protects the service providers from any of the usual legal risks of processing sensitive data so long as the parties follow the protocol. Similarly, collusion between BWWC and the Hariri Institute also exposes liability risks. Put simply: our threat model leverages the incentives provided by existing privacy law rather than trying to supplant it.

Read attacks against the service provider should yield no private input data, and MPC provides this guarantee. For write attacks, a tradeoff must be negotiated. While malicious-secure MPC can thwart write attacks, it does so by distinguishing between valid and invalid inputs (e.g., by requiring knowledge of the full set of participants ahead of time, by having a PKI or some other means of identifying which inputs come from which participants, and so on). In this work, supporting an a priori unknown number of users, not linking inputs to the original contributor, and a variety of usability requirements enumerated in Section II-D take precedence.

An implementation must protect the privacy of the data contributed by the contributors (apart from what is leaked by the aggregate output) against a semi-honest, static adversary capable of corrupting at most one party. Both data at rest and in transit may rely on hardness assumptions such as RSA [11].

Finally, we note that output privacy (informally speaking) increases with the number of contributors, as the output is an aggregate of the inputs (e.g., in the pathological case of a single contributor providing input, that input is necessarily leaked by the output). Thus, the simplicity and accessibility of the framework, as well as the comprehensibility of the underlying cryptographic tools, serve an important purpose (i.e., indirectly contributing to the overall security of the protocol by encouraging participation). As a lower bound, we also require that a minimum of five contributors provide data before the service provider allows the analyzer to compute an analytic. We currently do not quantify output leakage formally, though doing so within the framework of differential privacy presents interesting directions for future work.

D. Usability Requirements

A secure multi-party computation protocol only has value if multiple parties trust it and use it. In particular, our pay equity scenario involves individuals with a wide range of technical backgrounds utilizing computing resources that are outside of our control and are governed by a variety of organizational constraints. Therefore, our protocol and web service must satisfy many usability goals.

- *Comprehensibility*: To drive adoption, the secure MPC protocol must be simple for users to understand.
- *Auditability*: To inspire trust, the web service must have complete transparency, with open-source code to enable outside auditing. Additionally, the secure MPC protocol must never give sensitive data to the service, not even the aggregate output of the analysis.
- *Accessibility*: To minimize any hurdles that might discourage participation, the software must be easily and rapidly deployable, requiring no setup, specialized software, specialized hardware, or public Internet addresses for any of the contributors or the analyzer.

- *Simplicity*: The software must be usable within a relatively narrow time window by non-expert human contributors whose technical expertise may only include basic familiarity with spreadsheet applications and web browsing clients.
- *Asynchronicity*: Contributors only need to be online while entering their data, and analyzers only need to be online at no more than two points in the protocol: to start the process and to compute the analytic.
- *Idempotence*: contributors must be able to resubmit (i.e., update) their data if they discover the data they submitted was corrupted (either through human error, through a software application failure, or both).
- *Feedback*: Incorrect or malformed data from even one contributor destroys the value of the aggregate analytic. Hence, the web interface must proactively warn users about spurious data (see Figure 1).

III. Existing Off-the-shelf Tools

The past few years have seen several successful deployments of MPC [12]–[14]. Consequently, mature software frameworks are available [5], [6]. In this section, we highlight two natural "off-the-shelf" candidate solutions and discuss how they fell short of meeting the usability requirements outlined in Section II-D.

Viff: Viff [5], relying on linear-secret-sharing (LSS), is an open-source MPC framework implemented as a Python module on top of the popular networking framework Twisted. Viff overloads operators to allow the programmer to specify high-level arithmetic protocols (supporting the composition of sum, product, and comparison operations) without explicitly managing the underlying networking and cryptographic operations. Viff offers great flexibility in the number of parties (2 to n) and security guarantees (passive, active, with various corruption thresholds).

However, Viff unrealistically[1] requires all contributors, service providers, and analyzers to run the Viff software on mutually available servers. To overcome some of this burden, we extended Viff to run on two servers hosted by two separate service providers and to allow contributors to use a browser to submit data in secret-shared form.[2] The analyzer acts as one of the service providers and retrieves the results using Viff's built-in output mechanism.

Even so, Viff still fails to meet two usability requirements: asynchronicity (both service providers must be simultaneously online for the duration of the computation) and accessibility (BWWC lacks the technical expertise to configure a web server, obtain a web certificate, and install our Viff prototype and dependencies). We rejected the alternative of the Hariri Institute deploying a server on BWWC's behalf, as it places too much trust in the service provider.

[1]Apart from usability, the performance of an LSS-based scheme degrades as the number of computing parties grows.

[2]Our prototype implementation is available at https://github.com/n1v0lg/salary-equity-proto.git.

Sharemind: Another candidate framework is Sharemind, commercial software implementing a highly optimized LSS-based MPC scheme. Sharemind offers an extensive suite of privacy-preserving operations ranging from basic arithmetic to statistical and relational operations [15]. Contributors (respectively, analyzers) submit data to (resp., retrieve results from) the service providers via a web-based interface. Protocols are specified in SecreC [16], a domain-specific, C-like language. While Sharemind is building out support for two-party protocols [17], the core framework assumes three service providers and offers passive, honest-majority security.

We remark that Sharemind's complexity and closed-source design inhibit comprehension and auditing. Additionally, deployment requires advanced technical expertise and multiple service providers. The web-based platform Secure Survey [18] resolves the deployment issues but introduces legal concerns for potential contributors in the US. Built on top of Sharemind, Secure Survey lets contributors use a web browser to submit secret shares of their data to three Sharemind service providers hosted by Cybernetica, Partisia, and Alexandra Institute, who then perform the secure computation. However, with servers located outside of US jurisdiction, it may be unclear to potential contributors in our study what remedies are available if the service providers collude.

Lastly, we point out that neither Viff nor Sharemind provide formal security guarantees. Currently available secure computation frameworks that do (e.g., ObliVM [7] or Wysteria [9]) are research prototypes not presently developed for production-level deployment.

IV. Protocol Definition

The protocol ultimately developed for this application is a variant of a technique that allows multiple parties to collectively compute a sum of their own individual quantities without revealing those quantities to one another [19].

A. Drawbacks of the Existing Protocol

The naïve secure sum protocol requires all participants to coordinate with one another to pass data in sequence so that it visits each participant exactly once. From an implementation standpoint, this would require a relatively sophisticated software infrastructure involving multiple client/server applications that would all communicate with one another and maintain state throughout the duration of the computation, complicating the software implementation process and increasing the chance of runtime errors. The synchronization requirement would also require each participant to run the application for the duration of the computation, which may span days or weeks. Finally, the sequenced approach does not support idempotent updates from contributors; if even one participant makes an error and wishes to resubmit their data, the entire protocol would need to be restarted.

B. *Our Protocol*

Let G be an appropriate additive group such as $\mathbb{Z}/2^{64}\mathbb{Z}$ and distinguish each contributor using an index $i \in \{1, ..., n\}$. We call a single execution of the protocol a *session* and it proceeds in the following way:

(1) the analyzer initiates the process by generating a secret and public RSA key pair (s, p) and a unique session identifier $j \in \mathbb{N}$, submitting p to the service provider, and sending j to all the contributors;[3]

(2) each of the n contributors possesses a secret *data* value $d_i \in G$ and does the following at least once[4]:
   - (a) generate a secret *random mask* $m_i \in G$ and calculate the *masked data* $r_i = d_i + m_i$,
   - (b) send r_i to the service provider and retrieve p from the service provider to send back $c_i = \text{Enc}_p(m_i)$;

(3) the service provider computes the sum of the masked data values to obtain the aggregate masked data quantity $R = \sum_{i=1}^{n} r_i$;

(4) the analyzer then retrieves R and all the $c_1, \ldots, c_n$ from the service provider, computes $m_i = \text{Dec}_s(c_i)$ for all i, computes $M = \sum_{i=1}^{n} m_i$, and obtains the final result $R - M = \sum_{i=1}^{n} d_i$.

The service provider never sees the random masks because they are encrypted using the analyzer's public key, and the analyzer never sees the individual masked data values unless it violates its promise not to collude with the service provider.[5] Figure 1 illustrates an example deployment of the protocol for two contributors.

Our protocol guarantees that any malicious outside participant that can observe and permanently store *all* the communications between any and all participants will gain no information beyond the aggregate being computed. We exploit this feature of the protocol in the implementation: the server used for housing the data communicated between parties does not need to be secure and can be commodity hardware purchased from any third-party provider. The security of our protocol relies on RSA [20] or any equivalent public-key cryptographic protocol.

V. Client and Server Implementations

The purpose of the software application is to allow a group of non-expert participants to execute a session of the protocol defined in Section IV-B. In particular, the software application automates all portions of the protocol except the initiation of a session (which can be done with a single manual click), the distribution of the session identifier (which are simply delivered to participants via email), and the entry of participant data (participants must use the client-facing interface to paste or enter the data before submitting). Since realistic scenarios involve not one numeric quantity per participant but a collection or table of labeled quantities, the software application actually implements the protocol in parallel on multiple labelled fields within a table. Beyond these features, the software application was developed under the constraints already enumerated in Section II-D, which informed the design and implementation decisions for the application.

The two main components of the application are (1) the back-end server that acts as the service provider and as the delivery mechanism for the client-facing interfaces, and (2) the client-facing interactive interfaces to be used by the analyzer and participants. The server is implemented using the Node.js framework. Server-side data is stored within an instance of MongoDB , and the application interacts with that instance using the Mongoose module. SSL/TLS is supported using Let's Encrypt via the letsencrypt-express module [21]. The client-facing interface is implemented using JavaScript, employing the jQuery, Underscore.js, Handsontable, JSEncrypt [22], and spdy [23] libraries, in particular.

As both the server and client applications are authored using JavaScript, critical components such as the aggregate computation routines are shared between the components, reducing the likelihood of errors and facilitating maintenance and updates to the application. The actual fields of the table displayed to users are programmatically generated and can be modified in a given deployment. The JSEncrypt library is used to employ 1024 bit RSA encryption for encrypting the participant masks. The participant masks themselves are generated as arrays of pseudo-random numbers using the `window.crypto` or `window.msCrypto` object [24], which uses a high amount of entropy from the host operating system to ensure that any patterns that could be discovered in the generated numbers are minimized. Of course, it is worth noting that if a participant uses a compromised implementation of a web browser, the implementation of the object may not conform to the published specification.

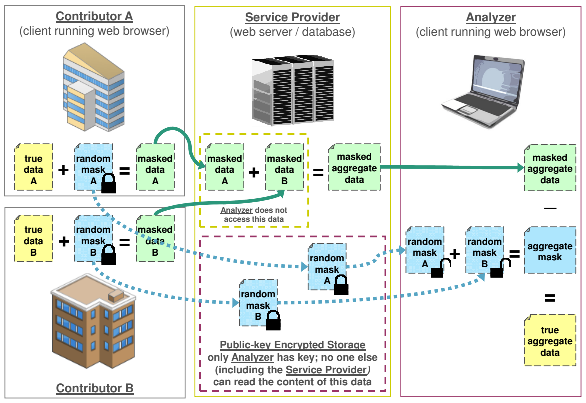
Figure 1 illustrates the web interface[6] as it appears within a web browser to each contributor. The participant interface provides a familiar spreadsheet table that an end-user can fill with data either manually or by pasting the data from another application. The email address is hashed on the client side and this hashed value is used only as an index into the server database, allowing each participant to submit more than once in a session (overwriting their previous submissions). There is also an analyzer interface that allows the analyzer to start a session, obtain a session identifier, and save a private RSA key to their local storage. There are also two other interfaces

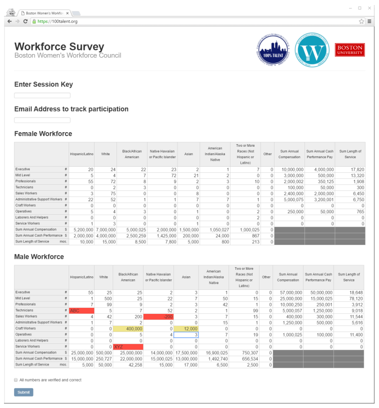
[3]The session identifier is only to allow distinct sessions, but it can serve another purpose: if no malicious agent possesses the session identifier, any data submitted by malicious agents will be ignored during the computation of the result.

[4]Each contributor can perform step (2) as many times as they wish before step (3) occurs; the operation they perform is idempotent if they always submit the same data.

[5]It is possible for the service provider to keep only a running total and immediately discard each submitted masked data value, but this merely makes it more difficult and not impossible for a malicious analyzer to obtain that data.

[6]The client application can be viewed at http://100talent.org.





Workforce Survey
Boston Women's Workforce Council

Enter Session Key

Email Address to track participation

Female Workforce

		Hispanic/Latino	White	Black/African American	Native Hawaiian or Pacific Islander	Asian	American Indian/Alaska Native	Two or More Races (Not Hispanic or Latino)	Other	Sum Annual Compensation	Sum Annual Cash Performance Pay	Sum Length of Service
Executive	#	20	24	22	23	2	1	7	0	10,000,000	4,000,000	17,820
Mid Level	#	5	4	7	72	21	2	0	0	3,000,000	500,000	13,320
Professionals	#	55	72	8	9	2	3	10	0	2,000,002	350,125	1,908
Technicians	#	0	2	3	0	0	0	0	0	100,000	50,000	300
Sales Workers	#	3	75	0	0	8	0	0	0	2,400,000	2,000,000	6,450
Administrative Support Workers	#	22	52	1	1	7	7	1	0	5,000,075	3,200,001	6,750
Craft Workers	#	0	0	0	0	0	0	0	0	0	0	0
Operatives	#	5	4	3	0	1	0	2	0	250,000	50,000	765
Laborers And Helpers	#	0	0	0	0	0	0	2	0	0	0	0
Service Workers	#	1	3	0	0	1	0	0	0	0	0	0
Sum Annual Compensation	$	5,200,000	7,000,000	5,000,025	2,000,000	1,500,000	1,050,027	1,000,025	0			
Sum Annual Cash Performance	$	2,000,000	4,000,000	2,500,259	1,425,000	200,000	24,000	867	0			
Sum Length of Service	mos.	10,000	15,000	8,500	7,800	5,000	800	213	0			

Male Workforce

		Hispanic/Latino	White	Black/African American	Native Hawaiian or Pacific Islander	Asian	American Indian/Alaska Native	Two or More Races (Not Hispanic or Latino)	Other	Sum Annual Compensation	Sum Annual Cash Performance Pay	Sum Length of Service
Executive	#	55	25	25	2	3	1	0	0	57,000,000	50,000,000	18,648
Mid Level	#	1	500	25	22	7	50	15	0	25,000,000	15,000,025	78,120
Professionals	#	7	99	9	2	3	42	1	0	10,000,250	250,001	3,912
Technicians	#	ABC	5	7	52	2	1	99	0	5,000,057	1,250,000	9,018
Sales Workers	#	4	42	200	-200	3	7	15	0	400,000	300,000	11,544
Administrative Support Workers	#	1	7	2	0	0	15	1	0	1,250,000	500,000	5,616
Craft Workers	#	0	0	400,000	0	12,000	0	0	0	0	0	0
Operatives	#	0	0	5	4	3	7	19	0	1,000,025	100,000	11,400
Laborers And Helpers	#	0	0	0	0	0	0	0	0	0	0	0
Service Workers	#	0	0	XYZ	0	0	0	0	0	0	0	0
Sum Annual Compensation	$	25,000,000	500,000	25,000,000	14,000,000	17,500,000	16,900,025	750,307	0			
Sum Annual Cash Performance	$	15,000,000	250,727	22,000,000	15,000,025	13,000,000	1,492,740	656,534	0			
Sum Length of Service	mos.	5,000	50,000	42,258	15,000	17,000	6,500	2,500	0			

☐ All numbers are verified and correct

Submit

Fig. 1. Diagram of a deployment of the protocol for two contributors used to explain the protocol to potential contributors (left) and the contributor web interface at http://100talent.org as it appears within a web browser with some user errors highlighted (right).

for the analyzer: a session tracker that displays how many participants submitted data, and a final unmasking and computation interface that requires them to supply the private key to the local client interface running in their browser in order to compute the final aggregate data. If too few participants have submitted their data, the service provider will not allow the analyzer to compute the final aggregate data. Once the final aggregate data is computed, it is displayed in the same familiar table format as the input table presented to individual participants.[7]

As discussed in Section II-C and IV-B, the ability to read any data on the server housing the database and delivering the client interfaces over the web to user browsers does not affect the security of the MPC protocol. Thus, all the interfaces are publicly accessible.

VI. Deployment and Future Vision

The protocol and software application described in this work were deployed successfully in 2015 and again in 2016 to collect compensation data for the purpose of pay equity analytics [25], [26]. The practical difficulties deploying the software application involved browser and operating system incompatibilities, human errors in using the application and the associated technical support activities, and the scheduling and duration of the data collection sessions. The simplicity of the protocol and its implementation greatly helped decision makers feel confident that they understood its operation and the security guarantees and contingencies that participation entailed. Hence, the participating employers (about 40 in the first collection and about 70 in the second) contributed data to the calculation of an aggregate analytic with confidence that their individual data would not be shared or released.

A more general-purpose platform enabling this type of privacy-preserving collective analysis has far reaching potential for public initiative research studies. Such a platform can allow for larger, more in-depth analyses and could accommodate other research needs where sensitive information from multiple parties must be collectively processed in order to identify trends, diagnose problems, or test hypotheses. Areas of application range from smart cities [27], to genomics [28], to cybersecurity [29], [30]. Based on our experiences, we believe this effort requires a collaboration between the security engineering, human-computer interaction, and social science communities.

First, we advocate that the security community combine the visions of secure multi-party computation and cloud computing. Secure MPC enables the most powerful computing entity to be the *least trusted* one. We envision a platform that provides strong computing and networking capabilities to "thin client" end users such that trust is inversely proportional to computing power.

Second, to achieve good usability, this platform should function using a star topology, where all contributors and analyzers use browser clients or mobile applications. Additionally, our experiences highlight the need for a general-purpose MPC platforms to deploy strong UI/UX features before deployment, since by its very nature MPC complicates detection of errors post-collection.

Third, deploying any such platform involves overcoming not only technical and logistical challenges, but also challenges that may be appropriate to address under the umbrella of social science studies encompassing law, organizational psychology, and behavioral economics. One particular experience we had deploying the application was that despite the guarantees provided by secure MPC, some participants still required that the analyzer and the service provider sign a non-disclosure agreement governing data submitted by those participants, even though this data contained no meaningful information content. In fact, contributors to secure MPC would benefit more from the creation of a different legal construct: a "non-collusion agreement" with enforceable civil penalties. More generally, we believe that social scientists can both contribute toward and receive great benefit from a general-purpose platform for the secure calculation of aggregate analytics.

[7]It is the responsibility of the analyzer to destroy their local copy of the private key after retrieving the result if this is the agreed-upon.

Acknowledgment

We would like to acknowledge all the members of the Boston Women's Workforce Council, and to thank in particular MaryRose Mazzola, Christina M. Knowles, and Katie A. Johnston, who led the efforts to organize participants and deploy the protocol as part of the 100% Talent: The Boston Women's Compact [31], [32] data collections. We also thank the Boston University Initiative on Cities (IOC), and in particular Executive Director Katherine Lusk, who brought this potential application of secure multi-party computation to our attention. The BWWC, the IOC, and several sponsors contributed funding to complete this work. Support was also provided in part by Smart-city Cloud-based Open Platform and Ecosystem (SCOPE), an NSF Division of Industrial Innovation and Partnerships PFI:BIC project under award #1430145, and by Modular Approach to Cloud Security (MACS), an NSF CISE CNS SaTC Frontier project under award #1414119. Finally, we must thank the many representatives from the participating employer organizations for their interest, diligence, and feedback throughout the orientation, testing, and deployment processes.

References

[1] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[2] A. C. Yao, "Protocols for secure computations," in *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*, ser. SFCS '82. Washington, DC, USA: IEEE Computer Society, 1982, pp. 160–164. [Online]. Available: http://dx.doi.org/10.1109/SFCS.1982.88

[3] O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game or A completeness theorem for protocols with honest majority," in *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA.* ACM, 1987, pp. 218–229.

[4] M. Ben-Or, S. Goldwasser, and A. Wigderson, "Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract)," in *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA.* ACM, 1988, pp. 1–10.

[5] "VIFF, the Virtual Ideal Functionality Framework," http://viff.dk/, [Accessed: August 15, 2015].

[6] D. Bogdanov, S. Laur, and J. Willemson, "Sharemind: A Framework for Fast Privacy-Preserving Computations," in *Proceedings of the 13th European Symposium on Research in Computer Security - ESORICS'08*, ser. Lecture Notes in Computer Science, S. Jajodia and J. Lopez, Eds., vol. 5283. Springer Berlin / Heidelberg, 2008, pp. 192–206.

[7] C. Liu, X. S. Wang, K. Nayak, Y. Huang, and E. Shi, "Oblivm: A programming framework for secure computation," in *IEEE S & P*, 2015.

[8] E. M. Songhori, S. U. Hussain, A.-R. Sadeghi, T. Schneider, and F. Koushanfar, "Tinygarble: Highly compressed and scalable sequential garbled circuits," in *Proceedings of the 2015 IEEE Symposium on Security and Privacy*, ser. SP '15. Washington, DC, USA: IEEE Computer Society, 2015, pp. 411–428. [Online]. Available: http://dx.doi.org/10.1109/SP.2015.32

[9] A. Rastogi, M. A. Hammer, and M. Hicks, "Wysteria: A programming language for generic, mixed-mode multiparty computations," in *Proceedings of the 2014 IEEE Symposium on Security and Privacy*, ser. SP '14. Washington, DC, USA: IEEE Computer Society, 2014, pp. 655–670. [Online]. Available: http://dx.doi.org/10.1109/SP.2014.48

[10] O. Goldreich, *The Foundations of Cryptography - Volume 2, Basic Applications.* Cambridge University Press, 2004.

[11] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978. [Online]. Available: http://doi.acm.org/10.1145/359340.359342

[12] P. Bogetoft, D. L. Christensen, I. Damgård, M. Geisler, T. Jakobsen, M. Krøigaard, J. D. Nielsen, J. B. Nielsen, K. Nielsen, J. Pagter, M. Schwartzbach, and T. Toft, "Financial cryptography and data security," R. Dingledine and P. Golle, Eds. Berlin, Heidelberg: Springer-Verlag, 2009, ch. Secure Multiparty Computation Goes Live, pp. 325–343. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-03549-4_20

[13] I. Damgård, K. Damgård, K. Nielsen, P. S. Nordholt, and T. Toft, "Confidential benchmarking based on multiparty computation," Cryptology ePrint Archive, Report 2015/1006, 2015, http://eprint.iacr.org/.

[14] D. Bogdanov, L. Kamm, B. Kubo, R. Rebane, V. Sokk, and R. Talviste, "Students and Taxes: a Privacy-Preserving Study Using Secure Computation," *PoPETs*, vol. 2016, no. 3, p. 117Ű135, 2016. [Online]. Available: http://www.degruyter.com/view/j/popets.2016.2016.issue-3/popets-2015-0019/popets-2016-0019.xml

[15] D. Bogdanov, L. Kamm, S. Laur, and V. Sokk, "Rmind: a tool for cryptographically secure statistical analysis," Cryptology ePrint Archive, Report 2014/512, 2014, http://eprint.iacr.org/.

[16] R. Rebane, "An integrated development environment for the SecreC programming language," Bachelor's thesis. University of Tartu, 2010.

[17] S. Siim, "A Comprehensive Protocol Suite for Secure Two-Party Computation," Master's thesis, Institute of Computer Science, University of Tartu, 2016.

[18] M. Vaht, "The Analysis and Design of a Privacy-Preserving Survey System," Master's thesis, Institute of Computer Science, University of Tartu, 2015.

[19] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M. Y. Zhu, "Tools for privacy preserving distributed data mining," *SIGKDD Explor. Newsl.*, vol. 4, no. 2, pp. 28–34, Dec. 2002. [Online]. Available: http://doi.acm.org/10.1145/772862.772867

[20] A. J. Menezes, S. A. Vanstone, and P. C. V. Oorschot, *Handbook of Applied Cryptography*, 1st ed. Boca Raton, FL, USA: CRC Press, Inc., 1996.

[21] "LetsEncrypt Express," https://github.com/Daplie/letsencrypt-express, [Accessed: June 25, 2016].

[22] "JSEncrypt," http://travistidwell.com/jsencrypt/, [Accessed: August 15, 2015].

[23] "spdy," https://www.npmjs.com/package/spdy, [Accessed: June 25, 2016].

[24] "Web Cryptography API," https://dvcs.w3.org/hg/webcrypto-api/raw-file/tip/spec/Overview.html, [Accessed: Aug. 15, 2015].

[25] J. Lipman, "Let's Expose the Gender Pay Gap," http://www.nytimes.com/2015/08/13/opinion/lets-expose-the-gender-pay-gap.html, [Accessed: August 15, 2015].

[26] R. Barlow, "Computational Thinking Breaks a Logjam," http://www.bu.edu/today/2015/computational-thinking-breaks-a-logjam/, [Accessed: August 15, 2015].

[27] "SCOPE: A Smart-city Cloud-based Open Platform and Ecosystem," https://www.bu.edu/hic/research/scope/, [Accessed: August 15, 2015].

[28] "NCI Cancer Genomics Cloud Pilots," https://cbiit.nci.nih.gov/ncip/nci-cancer-genomics-cloud-pilots/, [Accessed: August 15, 2015].

[29] L. Constantin, "IBM opens up its threat data as part of new security intelligence sharing platform," http://www.infoworld.com/article/2911154/security/ibm-opens-up-its-threat-data-as-part-of-new-security-intelligence-sharing-platform.html, [Accessed: August 15, 2015].

[30] "ThreatExchange," https://threatexchange.fb.com/, [Accessed: August 15, 2015].

[31] "100% Talent: The Boston Women's Compact," http://www.cityofboston.gov/women/workforce/compact.asp, [Accessed: August 15, 2015].

[32] "Boston: Closing the Wage Gap," http://www.cityofboston.gov/images_documents/Boston_Closing%20the%20Wage%20Gap_Interventions%20Report_tcm3-41353.pdf, [Accessed: August 15, 2015].