Secure Computation for Business Data

Contact: Gordon Long — glong@mitre.org

JSR-20-2E

November 2020

JASON
The MITRE Corporation
7515 Colshire Drive
McLean, Virginia 22102-7508
(703) 983-6997

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*
November 23, 2020

2. REPORT TYPE

3. DATES COVERED *(From - To)*

4. TITLE AND SUBTITLE

Secure Computation for Business Data

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

5d. PROJECT NUMBER
1319JAPM

5e. TASK NUMBER
PS

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

JASON Program Office
The MITRE Corporation
7515 Colshire Drive
McLean, Virginia 22102

8. PERFORMING ORGANIZATION REPORT NUMBER

JSR-20-2E

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

U.S. Census Bureau
4600 Silver Hill Road
Washington, DC 20233

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION A. Approved for public release. Distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The Census Bureau asked JASON to consider the use of secure computation technologies as a way of streamlining the collection and processing of business data used for economic analyses. Three use cases were considered: processing of confidential microdata held by Census for statistical purposes as a way to improve access for other agencies that do not have legal authority to access the microdata; collecting and processing business data as a potential substitute to traditional collection; linkage of records across datasets stewarded by diverse agencies without ingesting all the relevant data. JASON examined three approaches to secure computation: fully homomorphic encryption, secure hardware enclaves and multiparty computation. Of these three, the technology judged most appropriate for Census Bureau applications at this time is multiparty computation, but further development is required in order to fully meet Census Bureau requirements. JASON made several recommendations regarding future development of multiparty computation approaches. The most important of these is for the Census Bureau to engage in a series of pilot projects to fully evaluate the potential of multiparty computation in Census Bureau surveys. While secure computation technologies can improve the process of integrating diverse datasets to generate statistical products, some of the access issues are legal and not technical and so determining the full promise of such technologies will require further deliberation.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	b. ABSTRACT	c. THIS PAGE			Nick Orsini
					19b. TELEPHONE NUMBER *(include area code)* 301-763-6959

Contents

1 EXECUTIVE SUMMARY **1**
- 1.1 Findings . . . 8
- 1.2 Recommendations . . . 11

2 INTRODUCTION **13**
- 2.1 Overview of the Study and Charge . . . 13
- 2.2 Data Privacy vs. Inference Privacy . . . 18
- 2.3 Legal Aspects of Data Sharing . . . 20
- 2.4 Summary of the Study . . . 25
- 2.5 Overview of Report . . . 25

3 CENSUS BUREAU/BEA USE CASES **29**
- 3.1 Calculation of GDP and the National Accounts . . . 29
- 3.2 Use Case 1 - Income Statistics by Business Size . . . 32
- 3.3 Use Case 2 - Processing Business Data Directly . . . 36
- 3.4 Use Case 3 - Distributed Record Linkage . . . 42

4 TECHNOLOGIES FOR SECURE COMPUTATION **49**
- 4.1 Threat Models for Secure Computation . . . 50
- 4.2 Fully Homomorphic Encryption . . . 53
  - 4.2.1 Building homomorphic encryption . . . 54
  - 4.2.2 Ring learning with errors . . . 59
  - 4.2.3 FHE is promising but further progress is required . . . 60
- 4.3 Secure Multi-Party Computation (MPC) . . . 61
  - 4.3.1 MPC protocols . . . 61
  - 4.3.2 Trust models for MPC . . . 66
  - 4.3.3 Applications of MPC . . . 70
- 4.4 Trusted Execution Environments . . . 77
  - 4.4.1 Overview of Intel SGX . . . 79
  - 4.4.2 Security issues . . . 83
  - 4.4.3 Future approaches to secure enclaves . . . 90
- 4.5 Programming for Secure Computation . . . 92
  - 4.5.1 Circuit-based protocols . . . 92
  - 4.5.2 Secure enclaves . . . 94

5 CONCLUSIONS **97**
5.1 Mapping MPC Technologies to Census Bureau/BEA applications 97
5.1.1 Income statistics by business size 97
5.1.2 Processing business data 99
5.1.3 Distributed record linkage 100
5.1.4 Challenges for Secure Computation 100
5.2 Engendering Trust 102
5.3 Responses to the Sponsor's Questions 105
5.4 Findings 109
5.5 Recommendations 112

Abstract

The Census Bureau asked JASON to consider the use of secure computation technologies as a way of streamlining the collection and processing of business data used for economic analyses. Three use cases were considered: processing of confidential microdata held by Census for statistical purposes as a way to improve access for other agencies that do not have legal authority to access the microdata; collecting and processing business data as a potential substitute to traditional collection; linkage of records across datasets stewarded by diverse agencies without ingesting all the relevant data. JASON examined three approaches to secure computation: fully homomorphic encryption, secure hardware enclaves and multiparty computation. Of these three, the technology judged most appropriate for Census Bureau applications at this time is multiparty computation, but further development is required in order to fully meet Census Bureau requirements. JASON made several recommendations regarding future development of multiparty computation approaches. The most important of these is for the Census Bureau to engage in a series of pilot projects to fully evaluate the potential of multiparty computation in Census Bureau surveys. While secure computation technologies can improve the process of integrating diverse datasets to generate statistical products, some of the access issues are legal and not technical and so determining the full promise of such technologies will require further deliberation.

1 EXECUTIVE SUMMARY

The US government supports a decentralized Federal Statistical System of over 125 federal programs and agencies whose mission is to collect data and publish statistics for governmental decision making. Prominent examples are the Bureau of the Census, Bureau of Economic Analysis (BEA), Bureau of Labor Statistics (BLS), Bureau of Transportation Statistics (DOT) and the Statistical Information Service of the Internal Revenue Service (IRS). In order to produce their various statistical products, these agencies often must share data. For example, the BEA relies on data from the IRS, Census Bureau, and other agencies in order to compute important national economic indicators such as the Gross Domestic Product. The ability to share and link data among agencies is also critical to producing the various data products that provide valuable insights into the nation's economic and demographic posture and that inform future policies.

Sharing of data among these agencies is, however, also controlled by a variety of legal requirements. For example, Title 13 of the US Code allows the Census Bureau to collect data for itself and other agencies, but requires that the various elements of personal or business information collected by the Census Bureau (termed *microdata*) must be protected. Microdata are protected; they cannot be shared with other agencies, and can only be used for the purpose of creating statistics. Title 26 of the US Code protects sensitive Federal Tax Information (FTI) collected by the IRS. The Census Bureau and BEA require certain IRS data in order to produce their statistical products and so both agencies have access to various specific FTI but their levels of access are not the same. A number of formal agreements and procedures are in place between these agencies and the IRS that control the use of FTI. While such restrictions are important as they protect privacy, each time the BEA or the Census Bureau wishes to access elements of FTI to produce a new statistical product, they must engage in negotiations to establish that the statistics produced do not divulge any sensitive information and

are consistent with the requirements of Titles 13 and 26. Similar negotiations are required for any agency needing access to sensitive microdata.

One proposed approach to expediting the sharing of sensitive information used for statistical purposes is the use of secure computation. Secure computation describes a variety of techniques developed over the past several decades that enable computation to be performed on data, while keeping the inputs (and all sensitive intermediate results of a computation) private. Statistical agencies may be able to use secure computation technologies to jointly compute business or demographic statistics that require the linkage of potentially sensitive information without ever divulging any details of that information.

The application of secure computation may also enable improvements to the data collection process. For example, the Census Bureau engages in several surveys where businesses provide transactional data, which are essential in economic health assessments. While some data must be provided by law as part of the economic census that takes place every five years, the type and frequency of collection of data might be limited by the possible reticence of companies to share financially sensitive information owing to concerns over loss of competitive advantage through potential disclosure, as well as the burden inherent in selecting and preparing data for the survey. Using secure computation technologies, businesses could send sensitive data such as income statements or even balance sheets to a statistical agency with the assurance that their data could never be exposed directly (to the statistical agency or their competitors) and would only be analyzed or combined with other data for statistical purposes.

The Census Bureau asked JASON to consider the use of secure computation technologies for processing of business data for economic analyses. Three use cases were put forth for consideration:

Use Case 1 Processing confidential microdata held by the Census Bureau for statistical purposes. Secure computation technologies may enable improved access for other statistical agencies that do not have the legal authority to access and process the data.

Use Case 2 Collecting and processing business data as a potential substitute or supplement to the traditional collection of data by the Census Bureau for various business surveys. In addition to increasing the trust of business data providers in the confidentiality of Census Bureau data collection processes by improving security, an ancillary goal of adopting secure computation technologies here is to reduce the reporting burden of the respondent to the survey as well as provide more timely statistics.

Use Case 3 Linking records across data sets stewarded by diverse statistical agencies. Secure computation may allow the Census Bureau (and other statistical agencies) to link their respective databases and create new data products by performing statistical calculations on records common to those databases in a distributed fashion. At present, a series of agreements must be executed to engage in such linkage and the relevant data must be transported to the Census Bureau.

The Census Bureau asked JASON to examine two types of secure computation technologies: algorithmic secure computation, implemented in software using established encryption approaches, and techniques that take advantage of secure enclaves implemented in trusted hardware. The two algorithmic approaches to be considered are *fully homomorphic encryption* (FHE) and *multi-party computation* (MPC). In FHE, the data are encrypted and mathematical operations are performed directly on the encrypted data with decryption taking place only when the desired statistical result is obtained. This allows a computation to be outsourced to an untrusted server that performs the operations on encrypted data without ever learning about the input, intermediate values, or result of the computation. In

MPC, data providers (or their agents) participate in a protocol and collaboratively perform a computation to jointly compute a function on their combined data without any participant learning about the inputs of the other participants. Secure enclaves are implemented using trusted hardware that is used to isolate the program that runs on the sensitive data within the enclave, with encryption also used to protect the data outside of the enclave.

JASON was asked to provide a technical assessment of the proposed approaches and their effectiveness as well as to suggest a path for future development. JASON was briefed by the Census Bureau on three specific examples of the use cases under consideration:

Use Case 1 A proposal by the BEA to calculate aggregate wages, employment, gross economic output and gross domestic product but stratified by the size of the business contributing to the data. This would require access to FTI and Census Bureau data that the BEA currently does not have.

Use Case 2 The Commodity Flow Survey (CFS), undertaken jointly by the Census Bureau and the DOT to survey the movement of goods in the US and the various domestic modes of transportation used to deliver those goods. The CFS is undertaken every five years as part of the Economic Census and only limited data are shared. The objective here would be to have businesses provide more granular data with greater frequency, and automatically feed this data directly to the Census Bureau. Secure computation technologies would be used to protect sensitive aspects of the data, thus enhancing trust among the data providers while possibly reducing the burden on respondents by enabling more automated data collection.

Use Case 3 Distributed record linkage as exemplified by the Census Bureau product OnTheMap, a public use data product that connects employment locations to residence patterns. The data on employment come from state agen-

cies while the data for residence and business locations come from Census Bureau demographic data. The challenge would be to perform the linkage of these datasets in a distributed fashion so that no one agency need ingest all the databases required to perform the linkage.

At present, the Census Bureau acts as a centralized data collector and aggregator as well as a trusted data curator. For example, the Census Bureau is acting as a trusted data steward and curator for surveys like the CFS and OnTheMap, and so is able to collect the information that would be used in the absence of secure computation to execute the types of statistical computations exemplified by the above use cases. In examining the potential benefits of the various secure computation technologies, it is necessary to establish that these approaches provide the intended benefits relative to the approach currently in use by the Census Bureau. It is also important to note that the implementation of secure computation technologies in and of itself will only partly reduce the need to trust humans and processes. For example, attention must be paid to the potential for unintended disclosure of sensitive information in any query to be performed.

JASON examined fully homomorphic encryption (FHE). Such an approach is, on the surface, a very attractive way to achieve the secure computation goals mentioned above. FHE allows a computation to be outsourced to a potentially untrusted provider, while relying on strong encryption to prevent disclosure of sensitive information. FHE would be deployed in ways that allow multiple providers to input data into the computation, while limiting the output to one key holder. While the ability to compute on encrypted data is an impressive step forward, the techniques for achieving FHE add enormous computational overhead and are not appropriate for the data volumes associated with the proposed Census Bureau applications in the foreseeable future. Work continues on making FHE practical, but several breakthroughs are still required before it can be used for problems of the size and complexity of the Census Bureau use cases.

JASON also examined secure multi-party computation (MPC), which is a well-studied although not fully mature technology with some commercial deployments and demonstrated uses that match several aspects of the Census Bureau use cases. MPC can be used in a number of trust models, scenarios describing the trusted sharing and use of sensitive data, that may be appropriate for the types of calculations under consideration by the Census Bureau. JASON identified three such models and developed the following taxonomy for the purpose of this report:

Single Steward MPC In this model a single trusted data steward (e.g., the Census Bureau) owns all the data but processes it using at least two servers using MPC. Such an approach could be used for integration of data with differing legal restrictions for access as is the case in Use Case 1.

Delegated MPC In this model there are many data providers who use secret-sharing to split their input between two or more servers, who then use MPC to compute on that data. Such an approach could be used in Use Case 2 provided one or more trusted entities could be identified to participate along with the Census Bureau in the data aggregation step. In this model, no single compute server can reconstruct any sensitive inputs on its own, so there is only a risk of exposure if the server operators collude.

Joint Data Provider MPC In this model data providers, for example various statistical agencies, employ their respective information resources and operate their own servers to participate in a joint MPC computation with the Census Bureau such as record linkage. Such an approach would be appropriate for Use Case 3.

An issue of note is that the use of cryptographic protocols plus the overhead of network operations makes MPC expensive for general purpose computations. Several applications using MPC and processing data volumes relevant to the Census Bureau have been successfully demonstrated by using specially tuned

algorithms and protocols for problems such as private set intersection. There are additional issues that must be researched further such as the use of MPC for imputation of missing data, probabilistic record linkage where matches in data sets to be joined are inexact, and the infusion of noise into the statistical results so as to implement methods to prevent inference of data inputs from the outputs such as differential privacy. In addition, it is important to demonstrate that the output of a joint MPC computation does not leak sensitive information (e.g. microdata) as a result of the nature of the query. Thus, the function to be computed by the MPC protocol must undergo thorough review.

JASON also examined the potential for the use of secure hardware enclaves, focusing on the Intel SGX implementation, which is the one most widely available commercially. The design goals of a secure enclave are to provide protection for running programs on sensitive data even on a compromised host computer. The security properties for this approach rely on the correctness and tamper resistance of the hardware implementation; the security and correct implementations of the protocols used to attest to the validity and state of the enclave, and to transmit data to it securely; and the proper management of cryptographic keys used in the attestation. In addition, programs designed to run in an enclave must be carefully written so as to avoid information leakage due to side channels.

The Intel SGX design has suffered from repeated security issues, making leakage possible not only of sensitive data but also of cryptographic information allowing corruption of the attestation process. While SGX enclaves may provide some measure of protection in a public cloud setting, at the present time they do not provide the required security guarantees for Census Bureau applications.

1.1 Findings

The following are general findings:

1. Secure computation technologies are not a substitute for query and disclosure avoidance analysis. It is essential that any functions to be computed be reviewed for potential disclosure of sensitive information, and that all implementations be carefully reviewed for potential leaks. Additionally, secure computation cannot in and of itself be the determining factor in deciding if a query or calculation satisfies a required statutory benefit.

2. Secure computation technologies provide some opportunities for replacing trust in humans and processes with trust in technical solutions, but can only partially reduce the need to trust humans and processes. In any potential deployment, it is important to consider the required security and disclosure properties, and to understand who or what is responsible for different aspects of ensuring them.

Our finding on fully homomorphic encryption is as follows:

3. Fully Homomorphic Encryption (FHE) is unlikely to be of use for any Census Bureau application in the foreseeable future. The computational costs associated with FHE are prohibitive for any Census-scale computation, and, although there has been rapid progress in reducing computational costs over the past several years, major breakthroughs would be needed before FHE becomes practical for the types of applications currently under consideration by the Census Bureau.

The following are our findings on multi-party computation (MPC):

4. MPC is a well-studied although not fully mature technology with tools

available for building MPC applications, successful deployments, and well - understood security properties.

5. MPC can be used to compute any function expressible as a circuit securely, but the costs, dominated by network bandwidth, of general purpose MPC are high.

6. For some computations, sufficiently efficient MPC solutions are known that can scale to billions of inputs, including solutions for tabulation and private set intersection with aggregation.

7. Using MPC securely requires attention to implementation details. It can be used to eliminate the need to trust other participants in the computation, but does not absolve one of the need to review the disclosure risks of the function to be computed, or the need to trust one's own software and hardware.

8. By design, using MPC means that data inputs are not visible for human review during the computation. Any data editing procedures must either be fully automated or performed by the data providers or perhaps a trusted third party.

9. Imputation of missing data can be performed within MPC in principle, but further work will be required to determine whether it can be performed for applications at Census scale.

The following are our findings on use of secure enclaves, in particular, Intel SGX:

10. SGX is designed for a very specific threat model: running code on a host where the operating system could be compromised. In principle, it enables running code in the secure enclave without exposing any data under protection of the enclave to the compromised host.

11. SGX currently does not provide the promised protection, and has been vulnerable to known exploits for its entire history. Recent vulnerabilities not only allow leakage of data from a single enclave, but enable forged attestations of the enclave state, completely breaking the security premises of the SGX ecosystem.

12. If a sound and secure enclave could be implemented that satisfies the goals of SGX, it could allow for a self-contained program to be audited for desired disclosure properties, and for external users to verify and validate that this is the only program that will have access to data deemed sensitive.

13. Adapting a program to run securely within SGX (or other Trusted Execution Environments) requires specialized expertise, the absence of which increases the risk that sensitive information is leaked.

14. At present, SGX provides no clear advantage over the trusted data steward model for the use cases of interest to the Census Bureau.

15. New, potentially more secure designs for hardware enclaves are under development, but are not likely to become widely available within a five year time horizon.

The following are our findings on the use of secure computation technologies for applications of interest to the Census Bureau:

16. There are potential uses of MPC technologies for two types of Census Bureau applications:

    - The Delegated MPC model may provide an improved expectation of trust when the Census Bureau wants to collect and analyze data from providers who may be reluctant to participate owing to concerns of security and confidentiality.

- For applications where the Census Bureau and other agencies want to compute jointly on separately-stewarded data, the Joint Data Provider MPC model may provide a way to satisfy regulatory constraints with less reliance on inter-agency trust.

17. Three aspects of Census Bureau applications could pose challenges for the use of MPC technologies:

    - First, using MPC means that the data inputs are not visible for human review. Any data editing procedures must either be fully automated or performed by the data providers themselves. Any auditing of provided data or mechanisms to detect misbehaving data providers must also be fully automated, so algorithms used to aggregate such data must be resilient to inputs that may be provided maliciously.
    - Second, imputation of missing data can be performed within MPC in principle, but, depending on the specifics of the imputation method and the size of the data, it may be practically infeasible. Further research is required regarding this issue.
    - Third, archiving of secret-shared data is possible, but the protections afforded by MPC may be incompatible with regulatory archiving requirements. Archiving must either be performed in a way that preserves data privacy, thus requiring agreement among the data providers when the data are restored, or in a way that combines the data, thus giving up the privacy properties provided by MPC.

1.2 Recommendations

JASON's recommendations are as follows:

1. JASON recommends that the Census Bureau undertake with BEA a pilot study to consider the use of the Single Steward model of MPC in order

to link confidential microdata. The initial pilot should be done using data fully accessible to the Census Bureau to gain experience without any cross-organization complexities. The initial computation should be specifiable as a simple program (i.e., code fits on one page) that can be effectively reviewed for potential query disclosure. If successful, a subsequent pilot should be undertaken in partnership with an external data provider and a simple joint computation should be performed.

2. JASON recommends that the Census Bureau undertake a pilot study to consider the use of the Delegated MPC model in performing the Commodity Flow Survey. A key goal of this study should be to investigate whether such an approach could mitigate privacy concerns for businesses as regards sharing of confidential business data with the Census Bureau, enhance trust (in particular whether there are mutually trusted organizations that would partner with the Census Bureau in the MPC), and whether collecting data with privacy guarantees would reduce respondent burden by allowing the respondents to be less selective about the data they provide.

3. JASON recommends that the Census Bureau investigate tests for potential disclosure of sensitive information by a query so that certain types of queries can be formally specified with sufficient precision to implement as algorithms without the need for human review. Such automation will streamline the processing of queries for all MPC models as well the Single Steward model. Automating query approval would have substantial benefits if possible, but poses both regulatory and technical challenges that would require further study.

2 INTRODUCTION

2.1 Overview of the Study and Charge

The US government supports a decentralized Federal Statistical System of over 125 federal programs and agencies whose mission is to collect data and publish statistics for governmental decision making. Prominent examples are the Bureau of the Census, Bureau of Economic Analysis (BEA), Bureau of Labor Statistics (BLS), Bureau of Transportation Statistics (DOT) and the Statistics of Income Division of the Internal Revenue Service (IRS). In order to produce their various statistical products, these agencies often must share data. For example, the BEA relies on data from the IRS, Census Bureau, and other agencies in order to compute important national economic indicators such as the Gross Domestic Product. The ability to share and link data among agencies is also critical to producing the various data products that provide valuable insights into the nation's economic and demographic posture and that inform current and future policies. Further, the ability to link and analyze data from multiple sources can lead to additional insights into issues of national importance.

An example is the Opportunity Atlas [12], a joint endeavor between the Census Bureau and researchers at Harvard and Brown Universities to measure the average earnings at adulthood of children growing up in any given neighborhood of the US including also information about their race, gender and parental income. This information is displayed graphically (as exemplified in Figure 2-1) making it possible to correlate prospects of upward mobility by where a child grows up. Several sources of anonymized data are used to produce the atlas:

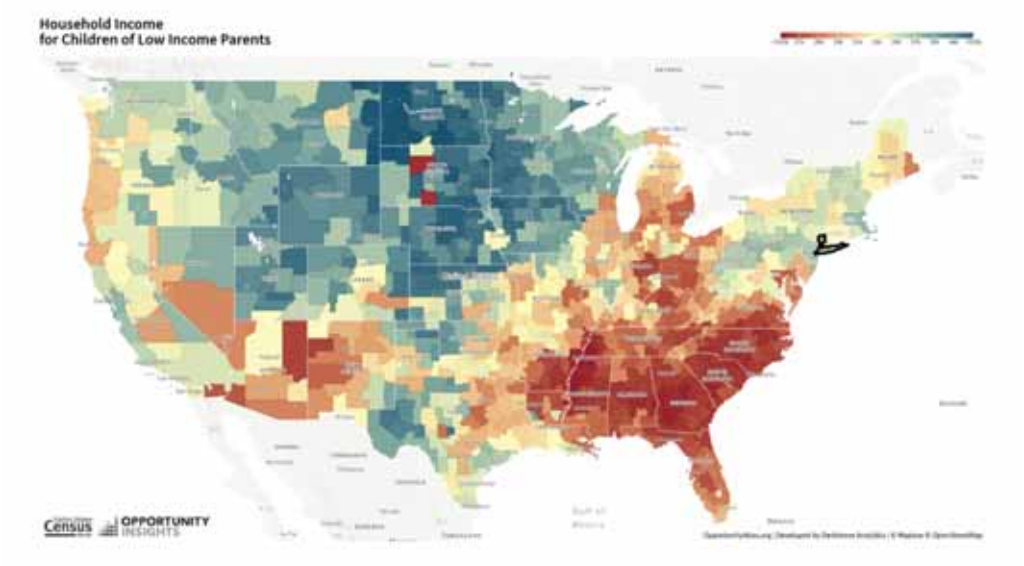


Figure 2-1: Illustration from the Opportunity Atlas [12] showing the household income earned at adulthood by children of low income parents as distributed over the US.

- The 2000 and 2010 Decennial Census short form,
- Data from federal income tax returns from the years 1989, 1994, 1995 and 1998-2015,
- the 2000 Decennial Census long form and the 2005-2015 American Community Surveys, and
- geographic information as determined by the location of Census tracts.

The notable aspect of this work is that diverse data sources are effectively linked across two statistical agencies, in this case, the Census Bureau and IRS, to produce important demographic information that can then be used in further investigations or in support of future programs. Chetty has expanded this work in developing an Economic Tracker [13] to investigate the decline in economic health across the country as a result of the Covid pandemic.

Sharing of data, as exemplified above, among various statistical agencies and the public is, however, also controlled by a variety of legal requirements that are discussed further below. For example, Title 13 of the US Code [81] allows the Census Bureau to collect data for itself and other agencies, but requires that the various elements of personal or business information collected by the Census Bureau (termed *microdata*) must be protected. Microdata cannot generally be shared with other agencies, and can only be used for the purpose of creating statistics.

One proposed approach to expediting the sharing of sensitive information ultimately used for statistical purposes is the use of secure computation. Secure computation describes a variety of techniques developed over the past several decades that enable computation to be performed on data, while keeping the inputs (and all sensitive intermediate results) private. Statistical agencies may be able to use secure computation technologies to jointly compute business or demographic statistics that require the linkage of potentially sensitive information without ever divulging any details of that information.

Beyond this is a larger vision of using secure computation as a way to encourage respondents to economic surveys (e.g. companies) to the surveys issued by various statistical agencies to provide more detailed data at a more rapid cadence. This would make it possible for the Census Bureau to perform economic surveys more frequently thus making it possible to report on the state of the economy with more recently acquired data. The use of secure computation would protect private inputs but allow the creation of aggregate statistics that do not disclose any sensitive information.

To investigate these issues further, the Census Bureau asked JASON to examine the use of secure computation for the purpose of economic data processing. In their Statement of Work, the Census Bureau provided three use cases for which the application of secure computation technologies may have benefit:

Use Case 1 Processing confidential microdata held by the Census Bureau for statistical purposes. In this scenario, the Census Bureau would submit confidential data under its control for processing via secure computation technologies that would then be used by the Census Bureau to aggregate the microdata into summary statistics. Such an approach could be used to create data products when contributing agencies do not have the legal authority to view or manipulate the data directly. The use of MPC would potentially make it possible to assemble such data products while respecting legal and privacy restrictions.

Use Case 2 Collecting and processing business data as a potential substitute or supplement to the traditional collection of data by the Census Bureau for various business surveys. In this scenario businesses would submit data such as balance sheets and income statements as is done now in economic censuses and other surveys. In addition, more granular data such as bills of lading or even general ledger data could be provided and aggregated by the Census Bureau into summary statistics. The use of MPC would make it possible for businesses to share a broader range of data and with greater frequency. An additional goal here is to expedite the process of data collection thus reducing respondent burden when such economic survey collections are performed.

Use Case 3 Linking records across data sets stewarded by diverse statistical agencies. The Census Bureau produces a number of data products that rely on the linkage of records originating from diverse sources. A barrier to performing such computations more frequently is the need at present to transfer the relevant databases to the Census Bureau facilities where they must be fully ingested to perform the required linkage. It would be desirable to perform the record linkage in a secure distributed fashion so that the entire databases need not be ingested. The use of MPC in principle allows

the Census Bureau to perform such record linkage and in the process create a richer set of data products providing additional insight into the nation's economy.

For each of the use cases described above there are several competing scenarios for performing the computation. One is the use of algorithmic approaches to secure multiparty computation (SMC) and here there are two main approaches to be considered : the first is the use of fully homomorphic encryption (FHE) wherein computations are performed directly on encrypted data with the final result available for decryption only by a designated holder of the decryption key. The second is Multiparty Computation (MPC) wherein multiple agents participate in a computation in which no party can learn the data inputs (other than their own) or sensitive intermediate aspects of the computation, but all participants learn the final result of the computation which can be shared among all the data providers. The other scenario for performing the computation is the use of secure hardware enclaves. Such enclaves are implemented in computer hardware to isolate a computation handling sensitive data from a potentially compromised operating system. A notable example is Software Guard Extensions (SGX) developed by the Intel Corporation.

JASON was asked to provide a technical assessment of the proposed approaches and their effectiveness for the use cases put forth above. JASON was also asked to suggest a technical path for future developments.

Finally JASON was asked to respond to the following questions from the Census Bureau:

- Is the Censsus Bureau researching technologies best suited to the purpose?
- What investments might make MPC technology operate at scale for a suite of business statistics?

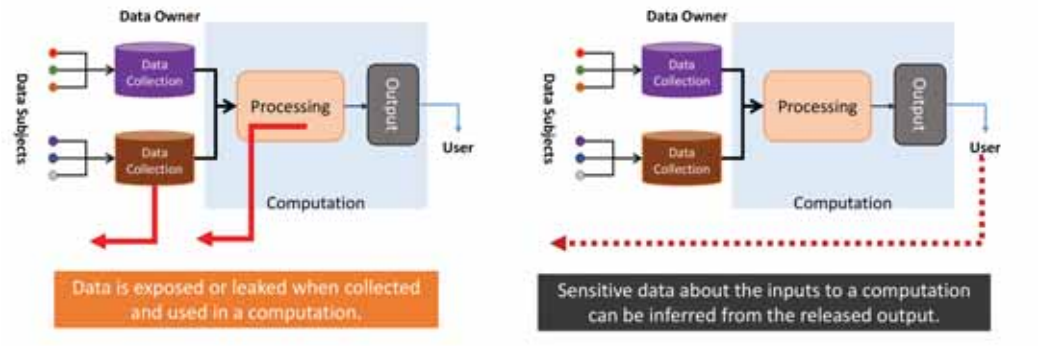


Figure 2-2: Data privacy vs. inference privacy. Data privacy refers to the protection of input data and sensitive intermediate results as a computation proceeds. Inference privacy refers to the inability to infer aspects of the data inputs from the computational output.

- Are there medium-range feasible SMC tools that could be used to enhance applications using record-level linkage without ingesting the full supplemental database?
- Would chaining Intel SGX enclaves in a commercial cloud environment support complex processing on a scale that meets Census Bureau needs?
- Does Secure Multiparty Computation (SMC) offer an opportunity to reduce burden on companies while continuing to provide the needed economic data?
- How does the Census Bureau build trust in SMC so that companies are willing to participate?
- Are there other conceptual approaches the Census Bureau should investigate?

2.2 Data Privacy vs. Inference Privacy

If one thinks of the operations required to create various statistics from confidential microdata as a type of computation, then the goal is to produce the statistics

without revealing any details of the inputs or any intermediate results of the computation that may be sensitive. This is known as *data privacy* in which sensitive aspects of the data should not be revealed as the computation proceeds. The technologies discussed in this report have as their objective data privacy.

In addition to data privacy, one must also consider *inference privacy*. Even though a computation is carried out securely, it might still be possible to infer information about the inputs. A trivial example is having the result of the computation be a simple invertible mapping of the input. A less trivial example is the release of many different statistics from a database. If more statistics are released than the number of elements in the data set then there is the possibility that the database records can be approximately reconstructed. Indeed, a result of Dinur and Nissim [21] known today as the Database Reconstruction Theorem shows that there exists a methodology to issue queries on a given database that will allow one to infer a database whose elements differ from the original in some number of elements. The number of elements that are not obtained correctly reduces as the number of queries increases. Thus the methodology asymptotically extracts all the elements of the private database.

A solution to this problem is to add properly calibrated noise to the statistics generated by a set of queries. This approach, put forth by Dwork and her colleagues [24] is known as Differential Privacy. The Census Bureau plans to apply Differential Privacy to the public data release of the 2020 Decennial Census and proposes to use the same methods to protect the output of the economic surveys under consideration here. We do not comment in this report on the use of Differential Privacy except to note that in applying it for the purpose of ensuring inference privacy one must also make use of secure computation in order to prevent leakage of sensitive microdata as it undergoes the various operations required to enforce Differential Privacy.

2.3 Legal Aspects of Data Sharing

Gaining a sense of the statutory requirements the Census Bureau and BEA must contend with is important in understanding some of the conditions that must be met with any data collection, sharing, and dissemination process. Such requirements are also relevant in considering the application of secure computation technologies. Federal statistical units and agencies must adhere to various rules that govern the collection, use, and protection of the data they collect. In the absence of agency or program-specific statutory guidance, the Confidential Information Protection and Statistical Efficiency Act (CIPSEA) applies [83]. The Census Bureau data collection activities, for example, are governed by U.S.C. Title 13 [81] whereas most of BEA's data collection is governed by U.S.C. Title 22 [82]. CIPSEA applies to both the Census Bureau and BEA with respect to data sharing authorities. These statutory requirements all lay out some important definitions. These include (paraphrased from Title 44 U.S.C. [83]):

- "**Statistical activities**":
    1. the collection, compilation, and analysis of demographic and economic data in order to "describ[e] or mak[e] estimates concerning the whole, or relevant groups" within the economy or society and
    2. the development of methods and resources to do so (e.g, models, measurement, frames).
- A "**statistical purpose**" is the impetus for or result of "statistical activities" (e.g., the "description, estimation, or analysis of the characteristics of groups, without identifying the individuals or organizations that comprise such groups").

Not just any statistical purpose will satisfy statutory requirements. For example, a Census Bureau statistical purpose must have a U.S.C. Title 13 benefit as described in DS002 [92]. For example

> "The project benefits the Census Bureau by analyzing changing demographic, social, or economic trends that affect Census Bureau programs, especially those that evaluate or hold promise of improving the quality of Census Bureau products."

A statistical purpose for the Statistical Information Service of the IRS also needs to satisfy a U.S.C. Title 13 benefit . The set of possible benefits for the IRS is a subset of that available for the Census Bureau, yet they are quite broad. For example a benefit that satisfies U.S.C. Title 13 use of IRS data is,

> "The project benefits the Census Bureau by leading to new or improved methodology to collect or tabulate data."

Several of the benefits outlined in DS002 [92] speak to improving the quality of data such as

> "The project benefits the Census Bureau by helping to understand or improve the quality of data the Census Bureau collects or acquires."

Secure computation is a natural fit for providing a U.S.C. Title 13 benefit as it offers the opportunity to provide a new methodology (process) for collecting data that will simultaneously improve the quality.

Another statutory twist exists for the Census Bureau's and BEA's access to Federal Tax Information (FTI). Under U.S.C. Title 26 [80] both of these agencies have access to these data but at different levels of granularity. For example, the BEA can only receive FTI on corporations for its statistical uses, whereas the

Census Bureau can receive FTI from the tax returns of a full range of business entities and individuals. This results in a restriction on the sharing of FTI between the Census Bureau and the BEA.

Finally, U.S.C. Title 44 [83] along with an agreement between the Census Bureau and the National Archives and Records Administration (NARA) [91] calls for the persistent archiving of the data the Census Bureau collects under U.S.C. Title 13 [81] and resulting statistical products, including the FTI they use for their statistical purposes.

To illustrate how this labyrinth of statutory requirements plays out in practice, we use the Census Bureau's creation of the synthetic data based on the Survey of Income and Program Participation (SIPP) [78] as an example.

SIPP is a household-base continuous series of national panel surveys carried out since 1983 and designed to capture national well-being based on the financial situation of households and individuals, as well as family dynamics and other socio-demographic information. Panelists (households) remain in the survey for approximately four years and new households replace households rotating off the panel on an ongoing basis. The data include a nationally-representative household sample that is generated from the Census Bureau's Master Address File. The SIPP Synthetic Beta (SSB) is based on the integration of microdata from SIPP panel surveys with administrative tax and benefit data [79]. The creation of SSB starts with a "Completed Gold Standard" data file by linking SIPP respondents' records directly to Social Security Administration (SSA)/Internal Revenue Service (IRS) Form W-2 records and SSA records of receipt of retirement and disability benefits. Once these data are linked, fully synthetic versions of these data are generated to create the SSB data product.

The Census Bureau is responsible for this data collection. Therefore, the goals and purpose of the Completed Gold Standard data file needs to align with

U.S.C. Title 13 benefits [92]. In this case, the benefits are providing more complete data products for relevant populations and enhancing data products as a result of linking SIPP data to SSA/IRS records. The Census Bureau is able to augment SIPP data with IRS data through their access granted to Federal Tax Information (FTI) under U.S.C. Title 26 [80]. However, the Census Bureau would have had to prepare a Predominant Purpose Statement to the IRS Statistics and Income Division to request the use of the FTI [90]. A U.S.C. Title 44 CIPSEA memorandum of understanding [83] would be required for use of SSA and such use would need to be consistent with at least one Title 13 benefit under DS002 [92]. To further create the SSB data product the purpose of the SSB would need to be consistent with the DS002 Policy on Title 13 Benefits matching benefits for both the Census Bureau and for the IRS.

Once the data for the Completed Gold Standard are collected and linked, only individuals granted Special Sworn Status (SSS) can have access to the confidential micro data. Based on the approval the Census Bureau obtained to create the Completed Gold Standard linked data, any statistical analyses the data can support would be allowed. But any release of results based on analyses of the confidential data would need to undergo U.S.C. Title 13 disclosure review [81] and, once released, would then become a public use data product.

To facilitate broader access to the data, the SSB project aimed to create high-fidelity synthetic versions of Completed Gold Standard data This was accomplished through a complex multiple imputation process [7]. Following data editing and imputations to fill out the original data in the Completed Gold Standard, multiple synthetic representations are created for all of the sensitive variables (as determined by the Census Bureau). The reason for multiple versions is to facilitate the proper variance estimates for statistics computed using the synthetic data. The synthetic data underwent a disclosure review to ensure U.S.C. Title 13 [81] confidentiality protection before it was made available.

Access to the SSB is not publicly available, but it does not require access to a Research Data Center or SSS. Access is provided through a restricted site – the Synthetic Data Sever. Users need to request approval from the Census Bureau to access the Synthetic Data Sever. The applications are reviewed for feasibility of completing the proposed project based on the SSB synthetic data fields available. Once approved, the analyses can be completed and publicly released without disclosure review because the synthetic micro data have already been through disclosure review and contain no sensitive microdata.

Researchers can request to have their analyses validated against the underlying Complete Gold Standard linked microdata. This can to be done without granting additional individual SSS to access the Completed Gold Standard data. Rather, the users submit their analysis codes after demonstrating they have run on the synthetic data successfully and Census Bureau staff will run the confidential data. The results will undergo disclosure review and if this is satisfied the results become public use products.

To complete the statutory picture, under U.S.C. Title 44 [91] the Census Bureau is responsible for a persistent archive of the Complete Gold Standard data, all versions of the data included in the Synthetic Data Server, and any Census Bureau approved public use data products.

The above discussion provides a picture of the statutory complexities faced by the Census Bureau in using diverse data sources to provide new statistical data products. In the case of SIPP, the Bureau used the generation of synthetic data as a way to broaden access. To apply this approach every time one wants to integrate various data sources is viewed by the Census Bureau as not being scalable. At the same time, acquiring the requisite special sworn status for all interested parties is also not scalable. Given this environment it is natural to ask if secure computation technologies could be used as a way of generating new data products while providing broader outside access to the finished aggregate statistics. In all likelihood,

the use of secure computation will not obviate the need to render statutory judgments on various proposals to integrate various data sources but it may provide guarantees that will make the required deliberations more straightforward.

2.4 Summary of the Study

JASON was introduced to the relevant issues through a set of presentations listed in Table 2-1. The briefers were experts both internal and external to the Census Bureau in areas such as economic surveys, business surveys, multiparty computation, secure enclaves, fully homomorphic encryption, differential privacy and federated learning. These talks were of high quality and were instrumental in educating JASON on these issues. Finally, the Census Bureau provided JASON with a set of reference materials.

In addition to the speakers provided by the Census Bureau, JASON also engaged Drs. Daniel Genkin and Srini Devadas to brief on various security issues associated with Intel SGX enclaves as well as proposals for future hardware design for secure enclaves. JASON also engaged in several telecons with Drs. John Abowd, Nick Orsini, Cavan Capps and Simson Garfinkel. JASON is grateful to all those who briefed JASON for their important contributions to this study.

2.5 Overview of Report

In order to provide the context for our findings and recommendations we describe in Section 3 the three use cases described above in more detail. Our purpose here is to categorize the type of data that are accessed and the associated aggregate statistics that are generated while highlighting those aspects of the computations that must be performed securely. We also provide an assessment of the computational burden in order to get some feel for the appropriateness of the use of secure computational technologies. Algorithmic approaches such as FHE and MPC can

[p!]

Table 2-1: Briefers for JASON Census Bureau study on secure computation.

Speaker	Title	Affiliation
Nick Orsini, John Abowd	Introduction and opening remarks	US Census Bureau
Daniel Goroff	SMC Technology Landscape	Sloan/ NSF
Jonan Chu, Daniel deGraaf	SGX	NSA
Kunal Talwar	Federated learning with integrated formal privacy	Apple Computer
Cedric Fournet, Vikas Bhatia	Protect your data in use leveraging Intel SGX and Azure Confidential Computing	Microsoft Corporation
Mayank Varia	SMC	Boston University
Marc Heiligman	Homomorphic Encryption Compilation and Techniques for Overhead Reduction	IARPA, DNI
David Archer	Practical secure sharing of sensitive data for better outcomes	Galois
Melissa Creech, Heather Madray	Legal requirements of 6103(j) and administrative requirements of compliance	US Census Bureau
David Wasshausen	Overview of BEA's national economic accounts	US Census Bureau
Brandy Yarborough	Business register introduction and overview	US Census Bureau
Tina Highfill, Erich Strassner	Use case 1: Linking Confidential Federal Microdata to Calculate Economic Statistics by Enterprise Size	US Census Bureau
Berin Linfors, Grant Degler, Christian Moscardi	Commodity Flow Survey	US Census Bureau
Cavan Capps	Use case 2: Collecting data using confidential techniques	US Census Bureau
Matt Graham	LODES/ OntheMap	US Census Bureau
Joe Near	MPC and SGX	Univ. Vermont
Daniel Genkin	SGX: Stuff Gets eXposed (briefing to JASON only)	Univ. Michigan
Srini Devadas	Towards secure high-performance computer architectures (briefing to JASON only	MIT
Simson Garfinkel, Cavan Capps	Telecon with JASON	US Census Bureau

add significant overhead to a computation and so it is important to understand whether the required computations can be performed in a reasonable time.

In Section 4, we assess the various technical approaches to performing the required computations securely. We begin with an assessment of fully homomorphic encryption. We continue with an assessment of secure hardware enclaves and conclude with an assessment of the use of multiparty computation (MPC). For each technology we focus on its strengths and weaknesses paying particular attention to whether a given technology could be appropriate for a given Census Bureau use case.

Finally, in Section 5 we conclude with a discussion of how one might map the various use cases to the MPC technologies discussed and a discussion of the various trust issues that must be addressed in applying secure computation technologies. Such technologies can alleviate some of the concerns associated with the need to compute results securely, but they are not a panacea. We close with our findings and recommendations and our responses to the questions posed by the Census Bureau.

This Page Intentionally Left Blank

3 CENSUS BUREAU/BEA USE CASES

In this section we summarize the three use cases briefed to JASON. Each use case represents an archetype of the data collection and patterns of computation. We begin with a brief overview of the national accounts produced by the BEA. These statistics provide a comprehensive view of U.S. production, consumption, investment, exports and imports, and income and saving. They are used to compute important economic measures such as the Gross Domestic Product. The description of the accounts serves as useful background for the first use case, generating income statistics by business size. The second use case is the collection of business data for the purpose of generating surveys of various aspects of economic activity; the exemplar here is the Commodity Flow Survey. Finally, the third use case is the use of record linkage of diverse datasets to generate new data products and insights into economic activity. The exemplar here is the assembly of the OnTheMap web application providing geographic analysis of where workers are employed relative to their residences. For each case we provide an overview, a description of the relevant computations and an estimate of the data volumes involved. This sets the stage for Section 4 in which we describe various secure technologies and how they might be used in performing the various data operations associated with each use case.

3.1 Calculation of GDP and the National Accounts

The National Income and Products Accounts (NIPA) are a series of accounting ledgers produced by the Bureau of Economic Analysis (BEA) that are used to assess the status of the national economy. The NIPA consists of seven summary accounts, with details provided in over three hundred supporting tables. The seven summary NIPAs constitute the accounting framework for estimating the value of production, distribution, consumption, and savings for the U.S. economy. Taken

together, the summary accounts comprise a double-entry system in which an expenditure (debit) in one sector is a receipt (credit) in the same sector or by another sector. This allows for double-entry bookkeeping, the accounting standard used in constructing the NIPAs. As an example, the most well-known account is the Domestic Income and Product account. The debit column of the account summarizes gross domestic product (GDP) measured by the expenditures approach, which is the sum of goods and services sold to final users. The credit column of the account summarizes gross domestic income (GDI) measured by the incomes approach, which is the income earned in production. By design, calculating economic activity either as production or as income yields the same answer, and is the basis for this double-entry account.

The NIPAs are based on source data obtained from a wide range of government entities including the Census Bureau, Bureau of Labor Statistics, Treasury Department, Office of Management and Budget, Agriculture Department, and Internal Revenue Service. Additionally, source data are obtained from a variety of private sector entities, such as trade associations. Once collected, source data have to be processed before they can be incorporated into the NIPAs. These calculations exercise a variety of statistical techniques for estimation including imputation, interpolation, extrapolation, and regression. Additionally, time series data often require seasonal adjustments and statistical calculations over moving time windows.

The NIPA estimates are updated on three time scales; every five years, annually, and quarterly. The update schedule is defined primarily by the availability of underlying source data. The comprehensive NIPA update takes places every five years. It is synchronous with the Census Bureau quinqennial U.S. Economic Census, which surveys the entire national economy. Participation of businesses in the quinqennial economic census is mandatory. Comprehensive updates also provide the opportunity to make definitional, statistical, and presentational changes that

improve and modernize the accounts. The comprehensive NIPA estimate, when it is produced, represents the most accurate and detailed picture of U.S. economic activity

Annual updates are carried out each summer and are based primarily on surveys drawn from samples of the businesses covered in the quinqennial economic census. Participation in annual surveys is mandatory for those businesses selected to participate. Such surveys include the Annual Survey of Manufacturers, the Annual Wholesale Trade Survey, the annual Retail Trade Survey, and the Annual Survey of State and Local Government Finances. While these surveys generally collect less detailed data than those collected in the quinqennial economic census, the annual revised NIPA estimates improve the quality of the picture of U.S. economic activity by incorporating data from the most recent calendar year.

Quarterly updates are based on quarterly and monthly surveys of businesses, for whom participation is voluntary. Such surveys include the Monthly Survey of Manufacturers' Shipments, Inventories and Orders, the Monthly Wholesale Trade Survey, and the Monthly Retail Trade and Food Services Survey. These surveys generally collect even less data than the annual surveys. Because participation is voluntary, these surveys also generate smaller sample sizes. Increasing participation rates and expanding the scope of the data in these monthly surveys would improve the quality of these quarterly NIPA updates, which are widely followed in the business and economics communities.

Applications based on methods of secure computation are expected to improve the privacy and security of source data that companies provide to the Census Bureau. It may also be that these applications will eventually reduce the burden associated with responding to survey data requests. By improving privacy and security, and reducing the burden of reporting, it may be possible to increase participation in the voluntary monthly surveys of businesses. This would greatly benefit the quarterly NIPA updates.

3.2 Use Case 1 - Income Statistics by Business Size

As a key example of Use Case 1 JASON was briefed by Erch Strassner and Tina Highfill of the BEA on a proposal to calculate aggregate wages, employment, gross output and gross domestic product by enterprise size. The BEA defines an enterprise as

> "a business, service, or membership organization consisting of one or more establishments under common, direct or indirect, ownership or control. It is the highest level of establishment aggregation. An enterprise may vary in composition, ranging from a single-establishment company to a complex family of parent and subsidiary companies (firms under common ownership or control)."

Statistics of income stratified by enterprise size would be of great value to decision makers as they would indicate in greater detail the various components that go into computing the Gross National Product and thus provide additional economic insight.

As indicated in Section 3.1, the BEA makes use of several data sources in producing its GDP and national income results. The amount and type of data that are shared is governed by various agreements among the agencies that provide this data. At present, BEA can develop some insight into the income distribution by enterprise size as it has access to the Census Bureau Statistics of US Businesses Survey (SUSB) [77]. From this survey, some aspects of the income distribution by business size can be inferred, but this approach is complicated by the fact that certain data are suppressed for privacy reasons. A much better approach would be to directly obtain IRS income data for sole proprietors, partnerships and corporations from the fields of their submitted tax returns and match the Employer Identification Number (EIN) or Social Security Number (SSN) on the return to

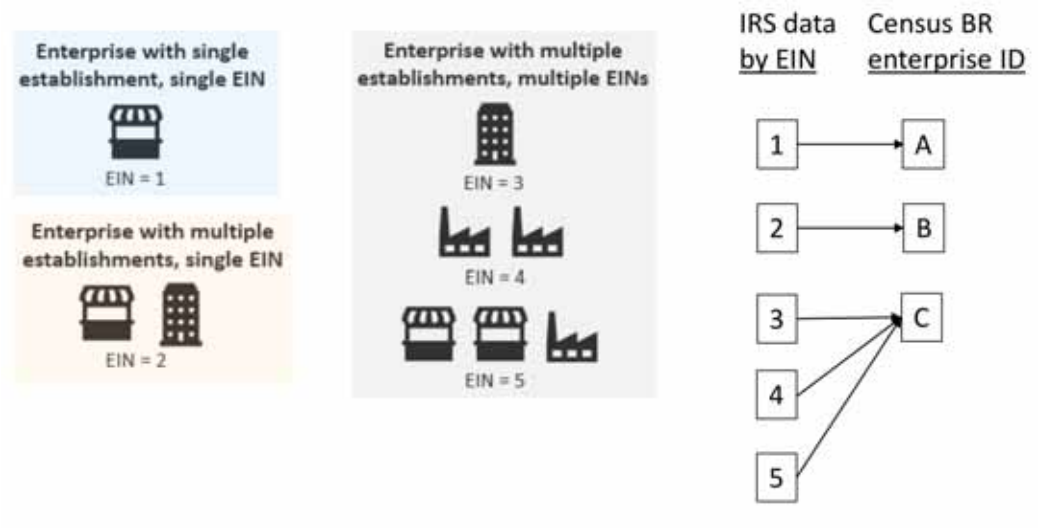


Figure 3-1: An example of the record linkage process in which the data for 3 enterprises consisting of 9 establishments and 5 EINs are linked to the Business Register. Graphic is from [73].

corresponding entries in a database the Census Bureau has developed for its economic surveys called the Business Register [75]. The Business Register contains structural information on various businesses like their size in terms of number of employees, the number of establishments, organization type (e.g., subsidiary or parent), industry classification, and operating data (e.g., receipts and employment). The Business Register is updated quarterly from multiple sources; the information is protected under both U.S.C. Titles 13 and 26 [81, 80].

In breaking out income data by business size, BEA would require additional information from the Statistics of Income Office of the IRS. In particular, for sole proprietors, BEA would require entries in Form 1040 Schedule C [68] for gross receipts, cost of goods sold, gross profit and gross income; for partnerships information from IRS Form 1065 [69] is required on receipts, profit income and salaries; similar data would be required from Form 1120 [70] for corporations. While such data are available to the Census Bureau through various agreements with the IRS, those agreements do not allow BEA at present to examine these entries.

IRS EIN	IRS form 1040, line item 26 (wages)	Census BR enterprise ID	Census BR enterprise employment	BEA enterprise size category
1	1,000,000	A	15	Very small
2	10,000,000	B	300	Medium
3	15,000,000	C	6,000	Large
4	10,000,000	C	6,000	Large
5	25,000,000	C	6,000	Large

BEA enterprise size category	IRS form 1040 wages by enterprise size	IRS form 1065 wages by enterprise size	IRS form 1120 wages by enterprise size
Very small	1,000,000	5,000,000	75,000,000
Small		25,000,000	750,000,000
Medium small		10,000,000	100,000,000
Medium	10,000,000	50,000,000	550,000,000
Large	50,000,000	500,000,000	7,500,000,000

Figure 3-2: A notional example of aggregation by industry and enterprise size of wages Once income data from the IRS can be linked with enterprise details from the Business Register [75] (top), it becomes possible to stratify business income by enterprise size (bottom). Graphic is from [73].

If this information were available, record linkage would be performed between the IRS returns and the Business Register to match corresponding EINs or SSNs. The size of a given establishment could then be determined and the income and other data could then be aggregated. The process is shown graphically in Figure 3-1. An example of the desired result is shown in Figure 3-2.

This use case illustrates an important issue. Title 26 of the US Code [80] protects sensitive Federal Tax Information (FTI) collected by the IRS. The Census Bureau and BEA require certain IRS data in order to produce their statistical products and so both agencies have access to various specific FTI but their levels of access are not the same. A number of formal agreements and procedures are in place between these agencies and the IRS that control the use of FTI. While

such restrictions are important as they protect privacy, each time the BEA or the Census Bureau wishes to access elements of FTI to produce a new statistical product, they must engage in negotiations to establish that the statistics produced do not divulge any sensitive information and are consistent with the requirements of Titles 13 and 26 [81, 80] and provide the required Title 13 statutory benefits while not releasing any sensitive information.

Similar negotiations are required for any agency needing access to sensitive microdata.

A key question is whether such negotiations could be expedited if it could be guaranteed that all details of the microdata involved in the computation could be protected using secure multiparty computation. It would still be necessary to check that the final results of the query also satisfied inference privacy but this could be achieved by applying the tools of differential privacy as part of the calculation.

This use case also illustrates a common pattern of computation in which microdata must be linked and then aggregated. The GDP and other NIPA calculations discussed in Section 3.1 correspond to a similar pattern in which confidential microdata from a number of sources are to be aggregated. In many cases, the relevant data reside within the Census Bureau, but any new computation still requires negotiation and a drafting of agreements among the various data stewards.

The sizes of the data sets in this particular use case are not particularly large by the standards of what is today termed "big data". The Census Bureau Business Register covers roughly 160,000 multi-establishment companies representing 1.8 million affiliated establishments, 5 million single establishment companies, and nearly 21 million non-employer businesses. As the BEA wishes to look at the establishments in the Business Register the total number of relevant EINs is about 6 million. The corresponding number of tax return entries is 23 for sole proprietors,

Figure 3-3: Left: Modes of transportation used in delivery of goods as summarized in the 2017 Commodity Flow Survey. Right: The top commodities shipped across the United States in billion-tons as summarized in the 2107 Commodity Flow Survey.

and 17 for partnerships and corporations. As will be discussed in Sections 4.3 and 4.4.1, this is well within the capabilities of secure technologies such as MPC and also secure hardware enclaves such as SGX.

3.3 Use Case 2 - Processing Business Data Directly

A key example of Use Case 2, the direct processing of business data, is the Commodity Flow Survey (CFS) [59]. The CFS is a joint effort of the Department of Transportation (DOT) Bureau of Transportation Statistics and the Census Bureau. The survey is the national source of data on commercial freight shipments as broken out by establishments in mining, manufacturing, wholesale goods, auxiliaries and selected retail and trade industries. The survey covers all 50 states and the District of Columbia (DC). It provides data on the type of goods, their origin and destination, distance shipped, ton-miles, value, weight, and the mode of transportation used for delivery. An overview of the principle modes of transportation used is shown in Figure 3-3 along with the main products shipped. As can be seen

from the figure, the primary mode of transportation used to ship commodities is by truck. It is for this reason the DOT, among other agencies has interest in these results. By using the tonnage and origin-destination information, processed through geographic software that provides commonly used highway routings for the commodities, the DOT gains knowledge of the level of utilization of various interstate and intrastate highways, demand for transportation facilities and services, energy use, safety risk, and environmental concerns. Using this information the DOT can prioritize future investments of highway funds. In addition, business owners and researchers use CFS data to identify trends in the movement of goods as well as spatial patterns of commodity and vehicle flows making it possible to forecast demands for the movement of goods, and determining needs for associated infrastructure and equipment. The CFS also provides important data for determining domestic supply chains although it does not provide full supply chain origin and destination data as the survey of shipments is limited to freight transportation in the US.

The CFS is performed every five years as part of the Economic Census. The data are collected quarterly during a survey year. At present, the Census Bureau is responsible for the design of the survey methodology and the data collection. The sampling frame for the survey is constructed in three steps:

- A set of establishments are selected from the Business Register. The North American Industry Classification System (NAICS) [89] provides a code for each business according to the type of activities it engages in. The survey is stratified by industry type, geography, and business size. Altogether about 100K businesses are selected.
- Each establishment is then assigned a week in a given quarter in which to report its shipments

Item F **SHIPMENT CHARACTERISTICS**

NOTE: Each line runs across pages 4 and 5. After entering column (I) data on page 4 for any line, continue with column (J) on page 5 for the same line.

Line No. (A)	Your Shipment ID Number (B)	Shipment Date (C) Month	Shipment Date (C) Day	Shipment value (excluding freight charges and excise taxes) in **whole dollars.** Estimates acceptable. (D)	Net Shipment Weight **in pounds.** Estimates acceptable. (E)	For shipments consisting of more than one commodity, report the code and description of the commodity that contributed the greatest weight of the shipment in columns (F) through (I): SCTG commodity code from accompanying booklet[3] (F)	Commodity Description[1] (G)	Is item in col. (G) Temperature controlled[2]? (Y/N) (H)	Is item in col (G) a hazardous material? Enter "UN" or "NA" number (I)	Continue with column (J) on page 5
Ex.1	123-5	4	26	224,235	4,840	34520	Mechanical machinery	Y		→
Ex.2	402H	4	26	1,375	50,125	20222	Sulfuric acid	N	1830	→
1										→
2										→
3										→
4										→
5										→
6										→
7										→
8										→

Figure 3-4: Sample from the CFS questionnaire. Graphic is from [50].

- Finally, the shipments to be reported are further limited by asking the establishment to sample the shipments in a given week. This is done because in some cases reporting of all shipments is not considered statistically necessary.

The latter step is sometimes considered to be burdensome by the respondents because it requires them to edit a list that otherwise could be reported directly by querying the responding shipper's database system. The geographic sampling is done over 132 CFS areas consisting of 84 metropolitan areas, 35 remainder-of-state areas and 13 whole states.

Once the sampling frame is finalized, each respondent receives a questionnaire, a sample of which is shown in Figure 3-4. For each shipment in the sample, the respondent must provide a number of descriptors including the Standard Classification of Transported Goods Code (SCTG) [85], the size of the shipment, its value, the mode of transportation, the destination zip code, etc. The question-

naires are for the most part filled out electronically, but the format is a spreadsheet that has the same format as the paper version, creating again some administrative burden for respondents. In fact, businesses report that it would be much easier to simply create a flat file with the categorized data for all shipments rather than the sampled set. The Census Bureau is currently working on providing a method to simply upload the data from the respondent using a web-based interface.

Because the internal representations of respondents used differ in their respective data handling systems, there are often missing entries and inconsistencies in the data received by the Census Bureau. For example, an SCTG code may be incorrectly assigned, the mode of transportation for the commodity may not be sensible, or the destination zip code may be incorrect. Such errors are dealt with using data editing and imputation. Census Bureau analysts must examine the errant entry and attempt to correct the errors via editing and missing data via imputation. In imputation, a "donor" shipment is examined that has similarity with the shipment under inspection and, if the match is sufficiently close, the missing or incorrect entries will be inferred from those of the donor shipment. Recently, the Census Bureau has investigated the use of machine learning as a way of correcting errors and inferring missing data with encouraging results. As will be discussed in Section 4.3, editing and imputation may pose challenges if one wishes to use any secure computation approaches approaches as direct access to unencrypted inputs is technically not allowed. Using such technologies may well require somewhat more elaborate computational approaches to insure consistency of respondent data.

Once the editing and imputation are completed, the data are weighted appropriately to account for the fact that only the data for a particular week in a given quarter has been collected whereas one is interested in statistical estimates for the entire year. Additional weighting is required to account for those businesses that did not respond as well as businesses that may have been created during the time

period of the survey. Finally an additional "noise" weighting is applied to enforce inference privacy so that respondents cannot be identified via inference on the final survey results.

Data users of the CFS are particularly interested in a key product that tabulates estimates of freight flows by origin, destination, commodity, and mode of transportation. This table has about 5M rows. But because the sampling methodology collects only about 6.4M shipments in total, many of the entries in this table are suppressed because there is insufficient data to estimate some of the entries. To reduce the number of suppressions, more data would be required.

To investigate how more data could be acquired in future surveys, the Census Bureau interviewed various representative shippers to see if they would be willing to provide more complete data more frequently. Shippers indicated that they were willing to provide more data, but the process for data collection had to be streamlined. To address this, the Census Bureau has initiated a pilot program in which shippers will deliver their data directly to the Census Bureau from their various business data systems via a new data upload tool. The proposed approach is shown graphically in Figure 3-5. This has the advantage that respondents need not expend time and effort on extracting and reformatting the required shipment data. Interestingly, the companies that were consulted do not have concerns over IT security. The Census Bureau is viewed as a trusted data steward and there is perhaps less sensitivity over sharing of shipment data.

The data volumes for the current CFS are again not particularly large. As discussed above, the Census Bureau currently surveys 100K establishments and collects about 7M shipping records. However, should the Census Bureau switch to a data collection mode in which companies provide shipping data say weekly over an entire fiscal year then the data volumes will become significantly larger, easily numbering in the billions. Such an approach will also require a different way of uploading the data to the Census Bureau. For the proposed increased

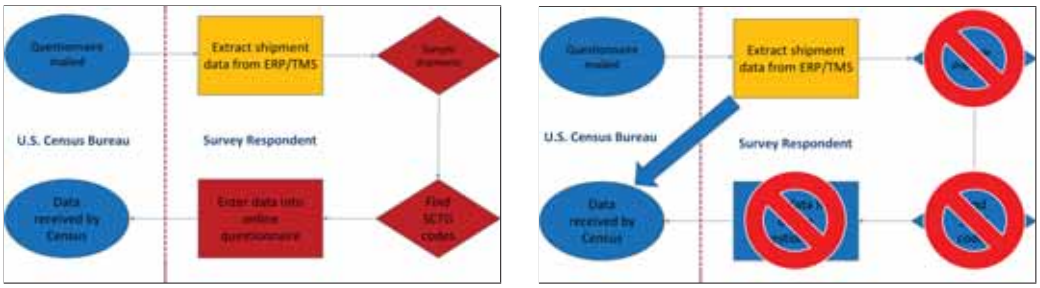


Figure 3-5: Left: previous approach to CFS data collection. Right: Proposed direct approach. Graphic is from [50].

data volumes and the possible use of secure computation technologies, it may not prove practical to edit or impute the entries. Instead, it may be necessary for the Census Bureau to work with participating companies and provide automated possibly web-based tools so that the data are made consistent prior to upload.

This use case illustrates a new paradigm of data collection in which businesses feed data directly to the Census Bureau on a regular basis making it possible to analyze more data and report economic statistics in a more timely fashion. For example, one might envision the communication of other economic data such as balance sheets, gross receipts, income taxes, etc. for other key surveys including those leading to estimates of such important statistics as GDP. In such cases however, the trust issues will be more acute and there will almost certainly be concerns over sensitivity of the data. If businesses agree to provide this additional data at a more rapid cadence they will require assurance that their data are processed securely so that external parties cannot view the data and that the final product will be only aggregate statistics appropriately protected for inference privacy via the applications of techniques for disclosure avoidance such as differential privacy. If such challenges can be overcome significantly deeper insight could be obtained into the US economy in a more timely way.

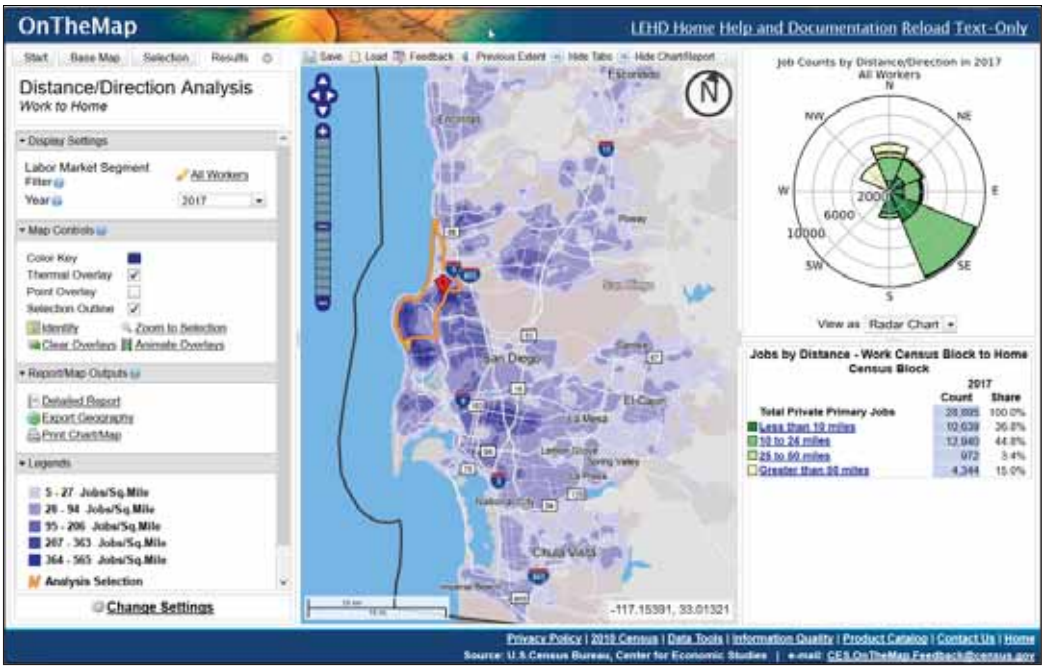


Figure 3-6: The use of OntheMap to display the distribution of jobs associated with the 92037 zip code (La Jolla, CA). The application also provides distance information for where people associated with these jobs live. Graphic is from [34].

3.4 Use Case 3 - Distributed Record Linkage

As a key example of Use Case 3, the Census Bureau briefed JASON on the processing steps required to assemble a public use query tool known as OnTheMap [76]. OnTheMap is a web-based application that shows where workers are employed and where they live. It can also be used to provide accompanying information on age, sex, earnings, industry distributions, race, ethnicity and educational attainment. An example of its use is shown in Figure 3-6 where the distribution of jobs in the La Jolla, CA area is shown. The application also provides a distance and direction distribution of where those people who hold the displayed jobs live.

The data sources for OnTheMap are assembled by a larger program known as LODES a double acronym standing for LEHD Origin-Destination Employment Statistics with LEHD standing for Longitudinal Employer Household Dy-

namics [88]. The LEHD program makes use of existing longitudinal job data from states and combines this with data from the decennial Census and American Community Surveys. The state data comes from a partnership among almost all 50 states, DC and Puerto Rico (with the exception of Arkansas, Mississippi, and Alaska). Under this partnership, states agree to share Unemployment Insurance earnings data with the Census Bureau. The Census Bureau then integrates this data with the Bureau of Labor Statistics Quarterly Census of Employment and Wages (QCEW) [84], and then links this personal and residence data from censuses and other surveys. From these data, it is then possible to generate statistics on employment, earnings, and job flows stratified by location, industry types and demography. Additional employment records are provided by the Office of Personal Management (OPM) to facilitate the generation of statistics for Federal employees. The assemblage of this data requires roughly 60 ongoing agreements among the Census Bureau and the contributing states and other agencies. These must be renewed periodically and are viewed as somewhat fragile. For example, JASON was briefed that OPM has recently decided not to provide job data for Federal employees associated with law enforcement, presumably for security reasons. This exemplifies the need to secure the input data using SMC and also to enforce inference privacy by making use of disclosure avoidance methods so as to protect job and residence information. The Census Bureau has achieved this recently through the use of Differential Privacy (cf. [53]). But further, it exemplifies the security concerns of linking such diverse data sources.

The core data required for assembly of the final origin-destination relation are processed quarterly from the following sources:

Unemployment insurance records Records are received quarterly from OPM and participating states and are used to create the universe of jobs. Information gathered includes earnings, job length and continuity;

Quarterly Census of Employment and Wages This is a survey sponsored by the BLS that provides the structure of the firms providing the employment data as well as characteristics associated with these establishments [84];

Business Register The Business Register is a database of establishments providing information on business location, organization type, business size. The data are provided by IRS, the Economic Census and other surveys [75];

Person records Records on employees are provided by the Social Security Administration, the Decennial Census [87], and the American Community Survey (ACS) [86]. These are used to create the universe of workers and their characteristics;

Residential records Records on addresses come from the ACS and Decennial Census, as well as administrative records from a number of agencies including HHS, HUD, IRS and the Postal Service

Geographic records These come from the Census Bureau's Geography Division as well as a private geocoding database;

Once these primary data sources are assembled, the Census Bureau creates a set of secondary input databases that are used in the final assembly of the LODES database and the OntheMap application. Creation of these databases requires linkages among the primary sources. Today this is done at the Census Bureau and requires the ingestion of the entire source input files. The various files required and the linkages established are shown in Figure 3-7. We list these secondary input databases and the required linkages below:

Employer characteristics file (ECF) The ECF creates the frame for businesses. It establishes the relationship between state business firms and establishments as listed in the Business Register. Because states only track state

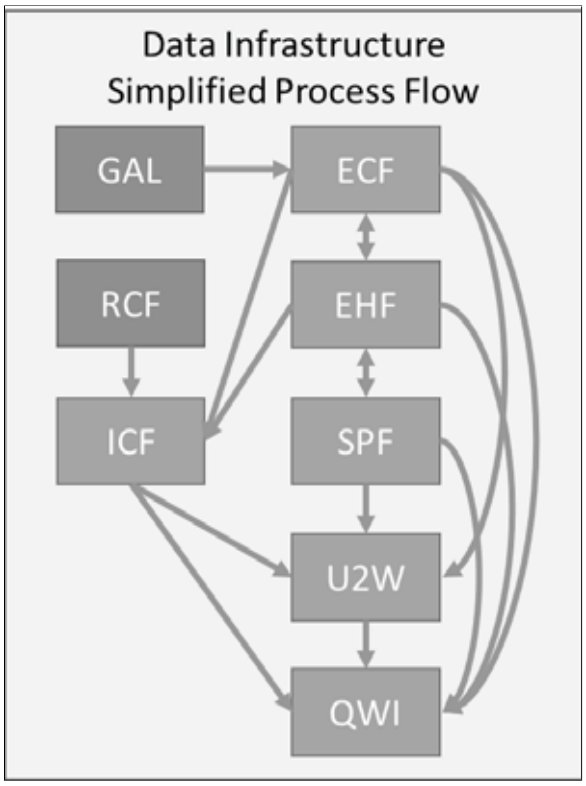


Figure 3-7: The data files that must be created for LODES and OnTheMap and their associated linkages. File abbreviations are defined in the text. Graphic is from [34].

businesses this is required to complete this relationship for multi-state businesses. In addition, each enterprise is assigned an enterprise ID. A link for location information is added by creating a foreign database key to the Geographic Address List (to be discussed below) and Census Block ID.

Employment history file (EHF) This file provides longitudinal salary information by year and quarter for each worker associated with a particular business.

Individual Characteristics File (ICF) This file links workers with personal data by assigning a private identification key (PIK) by linking with Social Security Information. If this is not possible, probabilistic record linkage is performed to obtain a best match.

Geocoded Address List (GAL) This file lists the locations for all establishments obtained from administrative data. It links via foreign keys to the Census Bureau Master Address File and geographic information from Census Block location data.

Residence Candidate File (RCF) This is the frame for residence addresses. Addresses are collected from administrative data and are tagged with a geocode.

Successor-Predecessor File (SPF) This file is used to identify when a business restructures. When this happens, salary records appear discontinuous. This file reestablishes the record continuity.

Unit to worker file (U2W) This files establishes the linkage between a job, a multi-establishment firm and the specific establishment where the employee actually reports to work. This is necessary because states do not report this information and it is necessary to obtain correct home-to-work distances. This is done via multiple imputation.

A final required product is the Quarterly Workforce Indicators (QWI) database providing local labor market statistics by industry, worker demographics, employer age and size and are an important source of research data in their own right. Once the ECF, QWI, ICF, and RCF are properly assembled from the input databases, the process of assembling the LODES data and ultimately the data for OnTheMap can be completed.

The data volumes associated with the processes listed above are large. For example, because LODES/OnTheMap is a longitudinal data product, the typical employment history file contains 5.5B records. A typical unit to worker file contains 10B records and the Quarterly Workforce Indicators file contains 2.8B records.

The process described above requires considerable coordination but also illustrates the potential of linking diverse data sources to create important information products that can provide valuable economic information to decision makers. A similar example cited earlier in the Introduction is the Opportunity Atlas [12]. In all cases, where linkage of diverse databases was required, various agreements had to be in place in order to access the data. A natural question is whether secure multiparty computation could be used so that the various agencies that provide the relevant datasets could participate in a computation the results of which would protect the privacy of the inputs as well as make use of disclosure avoidance to prevent inference of private data from the results of the computation. This will require secure distributed record linkage. This has been accomplished successfully but for the data volumes associated with LODES/OnTheMap state of the art algorithms are required as discussed further in Section 4.3.

Having examined the three use cases, we next discuss the various options for secure computation with a focus on the development and application of those technologies best suited to handle the associated data volumes and computational patterns.

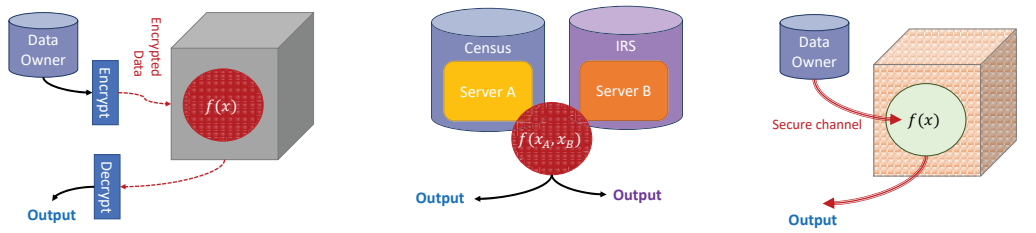
This Page Intentionally Left Blank

4 TECHNOLOGIES FOR SECURE COMPUTATION

Secure computing technologies enable computation to be performed on data without exposing that data. There are two main approaches that can be distinguished by their trust models: purely *cryptographic approaches*, in which the only computing device a data owner must trust is their own hardware and software, and all protections on data exposure are based solely on cryptographic mechanisms; and approaches using *trusted execution environments*, in which data owners place some trust in an externally-owned computing environment.

JASON reviewed the two main types of cryptographic approaches: *Fully Homomorphic Encryption* (FHE) (Section 4.2), in which an external computing service can operate on encrypted data, and *Secure Multi-Party Computation* (MPC) (Section 4.3), in which multiple parties execute a cryptographic protocol to cooperatively compute a function on their joint data; and the prevailing commercially-deployed trusted execution environment, Intel's *Software Guard Extensions* (Section 4.4). Figure 4-1 illustrates the three specific approaches discussed in this report.

Before getting into the specific technologies, Section 4.1 briefly discusses the main threat models that are considered in selecting and deploying secure computation technologies. We then discuss Fully Homomorphic Encryption. We then provide an extended discussion on Multiparty Computation as JASON feels that MPC technologies offer the best near-term opportunities for application to Census Bureau/BEA use cases. We then examine Trusted Execution Environments with a focus on Intel's SGX. In Section 4.5, we conclude this section with a discussion of how programs must be adapted to execute as secure computations.



Fully Homomorphic Encryption (FHE)
Server computes on encrypted data

Multi-Party Computation (MPC)
Multiple servers jointly compute obliviously

Trusted Execution Environments
Employ secure hardware (e.g., SGX)

Figure 4-1: Approaches to Secure Computation

4.1 Threat Models for Secure Computation

A *threat model* attempts to precisely capture the capabilities and goals of potential adversaries. All security mechanisms must be considered in light of the potential threats they are designed to thwart. For secure computation, we have one or more data owners that want to compute some function on their data. A potential adversary may want to learn sensitive information about that data, or may want to cause the computation to produce an incorrect output.

The two main threat models for secure computation are *semi-honest* and *fully malicious* security, which we define next. Those are extremal threat models, where the semi-honest model provides security guarantees only against very weak adversaries, and the fully malicious model provides strong guarantees against nearly any adversary. The trade-off is that protocols targeting stronger threat models can be much more expensive than protocols that only provide protections against weak adversaries. Many alternate threat models have been proposed, which provide trade-offs between these extremes.

There are a number of other concerns that may apply to secure computation protocols including *availability*, that is, the protocol always produces the output, and *fairness*, either everyone who should receive the function output does indeed

receive it, or no one does. We do not consider these further here, although for some applications these are also important security properties.

Semi-Honest Security A *semi-honest* adversary, also known as *honest but curious*, is a passive adversary who can observe the execution of a protocol, but cannot alter its execution. That is, a protocol participant that is corrupted by a semi-honest adversary still follows the protocol as specified, but the adversary attempts to learn as much as possible from the messages they observe during the protocol execution. Since semi-honest adversaries cannot alter the execution of the protocol, the only security property they can violate is confidentiality meaning that data, objects and resources are protected from unauthorized viewing and other access.

The semi-honest threat model makes strong assumptions about what an adversary may do. In particular, the adversary may not alter or inject any messages in the protocol, or change the actions of a corrupted protocol participant. The semi-honest adversary model may seem so weak as to be of no value; all the adversary can do is try to infer sensitive information from a normal protocol execution transcript. But in some settings, such as when servers used to execute a protocol are operated and controlled by large organizations with legal oversight, it is reasonable to assume adversaries are semi-honest.

Trusted Execution Environments, discussed in Section 4.4, can be used to establish a semi-honest threat model, where the adversary only observes properties of the computation that are visible outside a secure computing enclave and cannot modify the computation done within the enclave. Semi-honest protocols also often form the basis of protocols that provide stronger security; the first step to developing a protocol for a stronger threat model is to develop a semi-honest protocol.

Malicious Security A *fully malicious*, also known as an *active*, adversary is able to change how the corrupted participant behaves in arbitrary ways. By corrupted participant we mean here that the fully malicious adversary is corrupting the results seen by the participant. A malicious adversary has all the capabilities of a semi-honest adversary and can perform the same analyses of the protocol execution, but, in addition, they can alter any of the actions taken by the corrupted participants. Malicious adversaries can manipulate the contents of any messages in the protocol that are sent to the honest participants, and can alter the computations it performs however it wants to achieve its adversarial goals.

Since an active adversary can alter the execution, both confidentiality and correctness requirements must be ensured. For example, a malicious adversary may be able to alter the function that is computed by a secure computation protocol so that the output reveals more information about the other parties' sensitive inputs than the intended function would.

There are several methods known for transforming semi-honest protocols into maliciously secure ones. The two main approaches are requiring all protocol participants to provide a zero knowledge proof that confirms they have followed the protocol as expected [31]; the other is to use redundancy with randomly-selected openings, as in cut-and-choose protocols [11, 72, 100]. Both methods have high overhead for general-purpose protocols, and providing security against fully malicious potential adversaries can incur substantially more overhead than is required for semi-honest threat models.

Covert Security An alternative to malicious security, where a protocol must prove that the probability that an active adversary could violate the desired security guarantees is negligible in some security parameter that controls key sizes, is *covert security* [4], where the protocol must ensure that the adversary has a reasonable probability of getting caught. For many deployments of secure com-

putation where there are severe legal and financial penalties for an adversary who is caught behaving dishonestly, covert security provides a good trade-off between the semi-honest security, with its strong assumptions about limits on adversarial capability, and fully malicious security, that requires too much overhead for many applications. An example threat model for covert security is the *covert with public verifiability* model [3] which is satisfied if a protocol ensures that if the adversary cheats it will be detected with some probability (say $1/2$), and when cheating is detect an honest party will be able to produce an unforgeable and publicly-verifiable proof that establishes the cheating without revealing any of the sensitive data of the honest participants. For many settings, including general-purpose MPC protocols, protocols that provide covert security with public verifiability can be derived from semi-honest protocols with only a slight increase in execution cost [44].

4.2 Fully Homomorphic Encryption

The goal of Fully Homomorphic Encryption (FHE) [2] is to allow the execution of arbitrary programs on encrypted (confidential) data, while revealing no information about that data during the execution of the program. More formally, given a secret key $\boldsymbol{s}$, input $\boldsymbol{m}$, and function f to compute, a *homomorphic encryption system* provides an encryption method, E_s, a decryption method, D_s, and a way to transform f into f' such that $D_s(f'(E_s(\boldsymbol{m})) = f(\boldsymbol{m})$. The security goals are achieved if it is possible to compute f' with no knowledge of $\boldsymbol{s}$, and $E_s(\boldsymbol{m})$ provides no semantic information about the sensitive input $\boldsymbol{m}$. If the system can support any finite function as f, it is a *fully homomorphic encryption* system.

The idea of homomorphic encryption came from a proposal for privacy homomorphisms [65] and an observation that the RSA cryptosystem that depends on the multiplication of large integers [66] is homomorphic under multiplication since $x^e y^e = (xy)^e \pmod{n}$. Since then, several other cryptosystems such as the El Gamal system [25] were shown to be homomorphic under multiplication, with

other cryptosystems such as the Paillier scheme [61] exhibiting homomorphism under addition.

For many years, it was debated whether these partially homomorphic cryptosystems could be made fully homomorphic. That is, does there exist a cryptosystem that is homomorphic under both addition and multiplication? If so, then it would be possible using these operations (which correspond to logical AND and OR) to compute any finite function over encrypted data[1]. The first credible candidate for a fully homomorphic cryptosystem was the lattice-based system by Gentry [29]. It remains an open question what assumptions are needed to provide security guarantees for fully homomorphic encryption, and to make such systems practical.

Using FHE, it would be possible to outsource a computation processing sensitive data to a cloud provider. Even if the remote computer is corrupted in some way, all data and intermediate results are still protected as only the final result, which could be publicly shared, would emerge after decryption. The idea is depicted in Figure 4-2.

4.2.1 Building homomorphic encryption

The original proposal of Gentry used the theory of lattices to develop a scheme for FHE. Guided by the work of Micciancio [54], we next provide a summary of the basic ideas. Lattice cryptography is based on the difficulty of solving the following problem as indicated in Figure 4-3. Let A denote an $m \times n$ matrix of random integers modulo some prime number q which is of size that is polynomial in n, the size of a secret key, and let $\boldsymbol{s}$ denote the secret key, a n-dimensional vector of random integers modulo q. Consider solving the linear system

$$A\boldsymbol{s} = \boldsymbol{b},$$

[1] By finite function we mean here a function expressible as a finite Boolean circuit.

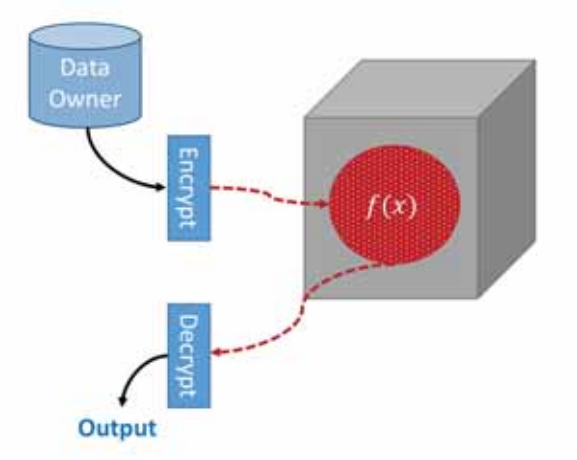


Figure 4-2: In fully homomorphic encryption a computation is outsourced to a remote platform and all intermediate manipulations on the data and intermediate results are performed on the encrypted data. The computation produces an encrypted result which is returned to the originator who can decrypt it to produce the function's output.

where $\boldsymbol{b}$ is simply the product of the matrix and vector. Given $\boldsymbol{b}$ and A it is simple to recover $\boldsymbol{s}$ by Gaussian elimination. Suppose, however, we add to the result $A\boldsymbol{s}$ some random noise $\boldsymbol{e}$ which is an m-dimensional vector of integers $\boldsymbol{e}$ sampled from a random distribution (usually normal) with zero mean and bounded by a number meant to be small relative to n, say $\beta = O(\sqrt{n})$, and call this result $\boldsymbol{b}$. In this case, one must solve the following system for $\boldsymbol{s}$

$$A\boldsymbol{s} + \boldsymbol{e} = \boldsymbol{b} \pmod{q}.$$

This small modification makes solving the system much harder and is known as the *Learning With Errors* (LWE) problem. The best known algorithms for solving this problem run in exponential time. Interestingly, this problem is also thought to be hard for a quantum computer, making lattice based cryptography an attractive alternative if quantum computers become available. This hardness property also makes the LWE problem an effective one-way function. Given A and $\boldsymbol{b}$, it is hard to deduce $\boldsymbol{s}$ and $\boldsymbol{e}$ but given $\boldsymbol{s}$ it is easy to deduce $\boldsymbol{b}$. Another interesting property is that the vector $\boldsymbol{b}$ is indistinguishable from a uniform distribution over the integers

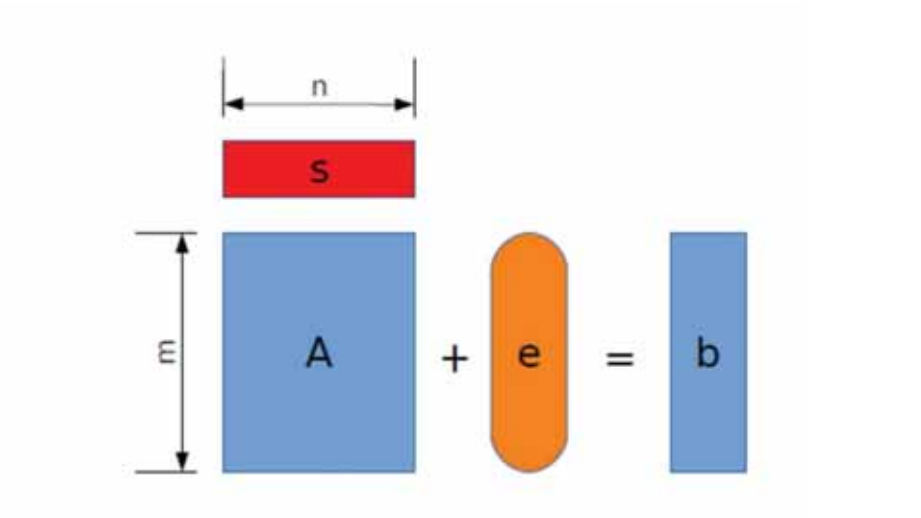


Figure 4-3: Graphical depiction of the encryption process in FHE based on learning with errors. The encryption takes place by multiplying a random integer matrix of dimension $m \times n$ by a secret vector containing randomly chosen integers of length n. A small error vector is then added to the result to produce the vector $\boldsymbol{b}$. This vector is then added to a message to be encrypted acting as a one time pad. Graphic is from [54].

$(\mathrm{mod}\ q)$. This makes the vector $\boldsymbol{b}$ usable as a type of one time pad if one wants to encrypt a message $\boldsymbol{m}$.

Encryption in the scheme looks as follows. Let $\boldsymbol{s}$ be a secret key of dimension n, let A be the random integer matrix, and let $\boldsymbol{e}$ be the small random noise discussed above. The encryption of a message $\boldsymbol{m}$ looks as follows

$$E_{\boldsymbol{s}}(\boldsymbol{m}; [A, \boldsymbol{e}]) \equiv A\boldsymbol{s} + \boldsymbol{e} + \boldsymbol{m}. \pmod q$$

Note that the ciphertext here is the combination A and the encryption $\boldsymbol{b} + \boldsymbol{m}$. This expands the size of the encrypted message considerably leading to questions regarding the practicality of such a scheme. We address this concern below.

Decryption is accomplished by using the secret key $\boldsymbol{s}$ to compute $A\boldsymbol{s}$ and subtracting it from $\boldsymbol{b} + \boldsymbol{m}$. This leaves the message but perturbed with noise. Because the size of the noise is "small" the message is corrupted in its low order bits.

But this can be filtered out by scaling the perturbed message and then rounding thus recovering $\boldsymbol{m}$.

The important properties of this type of lattice based encryption however are that it can be made homomorphic relative to addition and multiplication of cipher texts. First it is easy to see that we can add ciphertexts as follows

$$[A_1, A_1\boldsymbol{s} + \boldsymbol{e}_1 + \boldsymbol{m}1] + [A_2, A_2\boldsymbol{s} + \boldsymbol{e}_2 + \boldsymbol{m}_2] =$$
$$[(A_1 + A_2), (A_1 + A_2)\boldsymbol{s} + (\boldsymbol{e}_1 + \boldsymbol{e}_2) + (\boldsymbol{m}_1 + \boldsymbol{m}_2)]; \pmod q$$

so this scheme is clearly homomorphic relative to addition. Note too, that under each addition, the noise also adds up. The same is true if you multiply the ciphertext by a constant. If the constant is too large it can increase the noise to the point where the message can no longer be decrypted reliably.

Less obvious is that this scheme is what cryptographers call circularly secure. It is easy to compute encryptions of linear functions of the secret key without revealing the secret key. The decryption algorithm is linear in the ciphertext but it is also linear in the secret key. For example, you can show that if $\boldsymbol{s}' = (-\boldsymbol{s}, 1)$ then

$$D_{\boldsymbol{s}'}(A\boldsymbol{b}) = \boldsymbol{m} + \boldsymbol{e}, \pmod q$$

and

$$D_{c\boldsymbol{s}'}(A\boldsymbol{b}) = c\boldsymbol{m} + c\boldsymbol{e}, \pmod q$$

where c is some small multiplicative constant.

Multiplication by a constant of arbitrary size is achieved by defining a more elaborate encryption function

$$E'[\boldsymbol{m}] \equiv (E[\boldsymbol{m}], E[2\boldsymbol{m}], E[4\boldsymbol{m}], \ldots, E[2^{\log q}\boldsymbol{m}]). \pmod q$$

This construction can be used to show that

$$cE'[\boldsymbol{m}] = E'[c\boldsymbol{m}]. \pmod q$$

Finally, the following encryption function

$$E''(c) \equiv E'(c\boldsymbol{s}') \text{ where } \boldsymbol{s}' = (\boldsymbol{s}, -1) \pmod{q}$$

can be shown to have all the desired properties one wants:

$$E''(\boldsymbol{m}_1) + E''(\boldsymbol{m}_2) = E''(\boldsymbol{m}_1 + \boldsymbol{m}_2), \pmod{q}$$
$$E''(\boldsymbol{m}_1)E''(\boldsymbol{m}_2) = E''(\boldsymbol{m}_1\boldsymbol{m}_2). \pmod{q}$$

This construction is not quite fully homomorphic. The encryption function E'' can evaluate any arithmetic circuit, but with each operation the noise will grow and so in this form the scheme can only evaluate small circuits. The following idea can be used to clean the noise. Let

$$\boldsymbol{d} = E_{\boldsymbol{s}}(\boldsymbol{m} \times (q/2) + \boldsymbol{e}),$$

and let

$$f_{\boldsymbol{s}}(\boldsymbol{d}) = \mathrm{msb}(D_{\boldsymbol{s}}(\boldsymbol{d})) \times (q/2) = \boldsymbol{m} \times (q/2),$$

where msb means extracting the most significant bits. This operation effectively cleans the noise, but it can't be used as is because it requires decrypting the message. Gentry's [29] brilliant observation was that this cleaning operation could also be performed homomorphically. It turns out that the function $f_{\boldsymbol{s}}$ evaluated on an encryption of the key $\boldsymbol{s}$ leads to the same result as returned by the decryption but encrypted under the key $\boldsymbol{s}$. Thus the output noise will depend on the operations required to decrypt and extract the most significant bits, but not on the accumulated noise $\boldsymbol{e}$. This idea is known as *bootstrapping* and makes it possible to evaluate circuits of any depth or essentially any function that can be computed from circuits. Note that this makes use of an assumption that the encryption function E'' is circularly secure. It has been shown that E and E' are circularly secure, but it has not been shown yet that E'' has this property although it is believed that it is a matter of time before this is verified.

4.2.2 Ring learning with errors

As discussed above, the FHE schemes based on the LWE problem require large key sizes. This is simply because one has to provide the vectors for the random matrix A as well as the secret key $\boldsymbol{s}$. It would be desirable both for reasons of memory and computational efficiency to reduce the key size to something linear in the size of the message. One way to do this is to assume there is structure in the columns of A [63]. One chooses the first column of A

$$\boldsymbol{a}_1 = (a_1, a_2, \ldots, a_m)^{\mathrm{T}}$$

uniformly but the remaining vectors are chosen according to the rule

$$\boldsymbol{a}_i = (a_i, \ldots, a_m, -a_1, \ldots, -a_{i-1})^{\mathrm{T}} \qquad i = 2, \ldots, n$$

Now there are only $O(m)$ elements. In addition because the matrix now has a circulant form, it is possible to significantly speed up the matrix-vector multiply using the fast Fourier transform. Mathematically, this construction replaces the group of integers $\mathbb{Z}_q^m$ with a polynomial ring $\mathbb{Z}_q[X]/ < X^m + 1 >$. Because these constructions use a ring, the problem is called *Ring Learning With Errors* (RLWE). It turns out these types of problems are also hard and RLWE is a promising candidate for efficient implementations of FHE

At present, the fastest implementation of FHE is the TFHE library [14]. TFHE is an open source C/C++ library available via github. With security parameters equivalent to 128 bits of security, TFHE can evaluate about 70 bootstrapped binary gates in one second on one core of an Intel Core i7 or i9 processor. This is, of course, quite slow relative to ordinary cleartext computation. Typically, billions of gates must be evaluated for statistical calculations and so there would be considerable latency in using FHE to the extent that the use of FHE today is not practical for Census Bureau applications.

It is difficult to say how well FHE will perform in the future [57] since it has been subject to constant improvement. This improvement has taken orders of magnitude off the time required for computations, but the fundamental issues remain. Part of the issue, apart from the number of terms to be evaluated, is the size of the integers used in the arithmetic. Most modern CPUs have 64-bits as a native type, and a few support 128-bits, far short of what it needed to effect FHE operations in hardware. DARPA has initiated a project called DPRIVE (Data Protection in Virtual Environments) [20] that aims to make FHE practical through hardware acceleration.

4.2.3 FHE is promising but further progress is required

To summarize, there are two primary concerns regarding the adoption of FHE. The first more minor concern is that it remains to be proven that the fully homomorphic operation E'' is circularly secure. It is believed that it is, but proving this is apparently much more difficult than proving the circular security of the simpler operations like E and E'. The second concern, and most widely discussed, is performance. Although there has been great progress in improving the performance of FHE since it was first described by Gentry [29], homomorphic operations remain extremely slow relative to arithmetic as performed in the clear on modern processors such that it is impractical today to apply FHE for all but the simplest programs. Work continues in this area and it is important for the Census Bureau to continue to track progress as an implementation of FHE that can handle arbitrary programs and large datasets at reasonable speed would be a very important step forward in secure computing.

4.3 Secure Multi-Party Computation (MPC)

Multi-party computation (MPC) enables two or more participants to perform a protocol that computes a function on their combined data, without revealing each individual's input to the other participants or leaking any information from intermediate results [26]. At the end of the protocol, the output of the function can be revealed to one or more participants in the protocol. The key difference between FHE and MPC is that while FHE allows a computation to be fully outsourced to an untrusted computing service, MPC requires active participation of data owners in a protocol that performs the computation. Participants in an MPC protocol must agree on the function to compute together, and each participant maintains control over their own input to that computation.

The original pioneering work on MPC goes back to Andrew Yao in the 1980's who formulated the first protocols. At that time this work was viewed as mostly of theoretical interest. Today, MPC is viewed as a well-studied but not fully mature technology. Over the past two decades, with the advent of faster computers and networks, it has been transformed from a theoretical curiosity to, in some applications, a practical tool, with industrial deployments and tools available for producing MPC protocols. General purpose MPC is still expensive for large computations and is still maturing, but special purpose protocols have been developed that provide efficient solutions to many tasks relevant to data aggregation and statistics. Next, we provide a high-level introduction to MPC protocols. Section 4.3.2 discusses different ways of using MPC. In Section 4.3.3, we survey some applications of MPC.

4.3.1 MPC protocols

At a high level, all MPC protocols can be viewed as a form of secret sharing. In some protocols, data is explicitly secret shared, such as XOR secret sharing

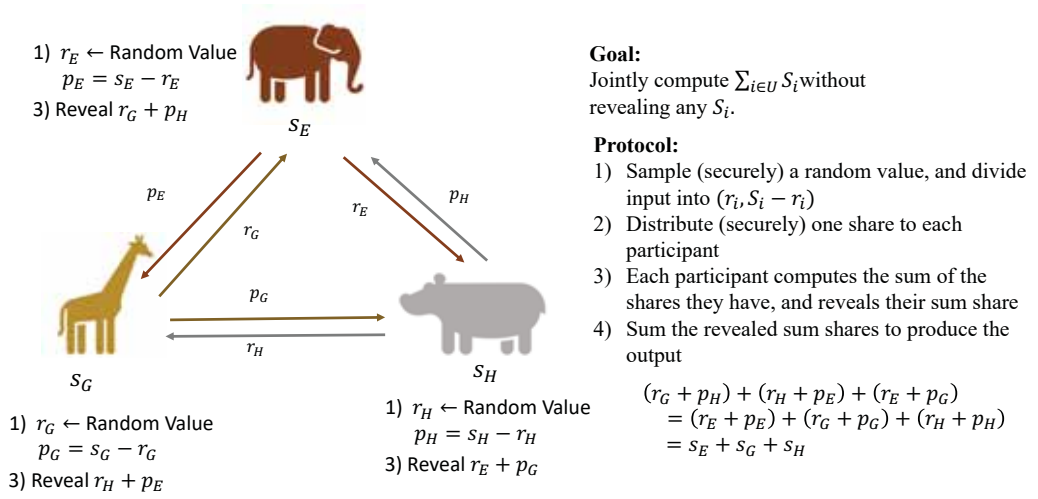


Figure 4-4: Example MPC protocol that uses additive secret sharing to jointly compute a sum.

in which a value x is split into two shares, x_R and a random mask R, which is randomly generated, with $x_R = x \oplus R$. Then, neither R nor x_R reveal anything about the semantic value of x, but the two shares can be combined to reconstruct $x = R \oplus x_R$. Another example is additive secret sharing, such as the protocol illustrated in Figure 4-4.

The other kind of secret sharing used in MPC protocols is where one participant holds encrypted data, $c = \mathrm{Enc}_k(x)$ and the other party holds the decryption key k. Like the XOR secret sharing, this has the property that if the encryption algorithm is secure, neither party by itself has meaningful information about x, but by combining their shares they can learn x. This type of secret-sharing protocol provides a simple way to do addition securely on shares and combine the results to produced the sum, but does not support other operations. There are several methods to extend this type of additive secret sharing approach to support multiplication (and hence, complete operations), but supporting both operations requires additional communication for each circuit layer. Examples of protocols of this type include Goldreich-Micali-Wigderson (GMW) [31, 30], and the Ben-Or, Goldwasser, and Wigderson [6] that builds upon Shamir secret sharing [71].

These costs can be moved to a pre-processing stage by using correlated randomness. For example, the communication needed to perform multiplications in the BGW protocol can be mostly moved into a preprocessing stage by using Beaver triples [5].

To provide an understanding of how general-purpose MPC protocols work, we describe one such protocol in a bit more detail. Yao's Garbled Circuits protocol (GC) is the most widely known MPC technique, and for general-purpose MPC, it is often the best performing option. Yao's protocol has low latency since the number of communication rounds does not scale with the size of the circuit, unlike protocols like GMW where a communication round is needed for each layer of the circuit.

The intuition behind Yao's protocol, illustrated in Figure 4-5, is that any finite function can be encoded as a circuit of simple Boolean operations (e.g., AND, OR, NOT), and those operations can be implemented as lookup tables. If the lookup tables can be evaluated obliviously, then the function can be computed obliviously. In Yao's protocol, one party, known as the *circuit generator*, generates a garbled circuit that computes the desired function, and the other participant, the *circuit evaluator*, evaluates the circuit obliviously. By oblivious computation we mean here that the circuit generator generates all the encryption keys used to produce the circuit, but learns nothing from its evaluation and the evaluator can evaluate the circuit, but because of the gate garbling, learns nothing about the inputs or intermediate results.

The encrypted lookup tables, known as *garbled tables*, consist of the encrypted entries that correspond to the output values of a given logic gate in randomly permuted order. Each virtual wire in the circuit is assigned two keys by the circuit generator, known as their *wire labels*, each one to represent each semantic value the wire could hold. The mapping is not known to the evaluator, and the evaluator only knows the wire labels that it learns by decrypting the row corre-

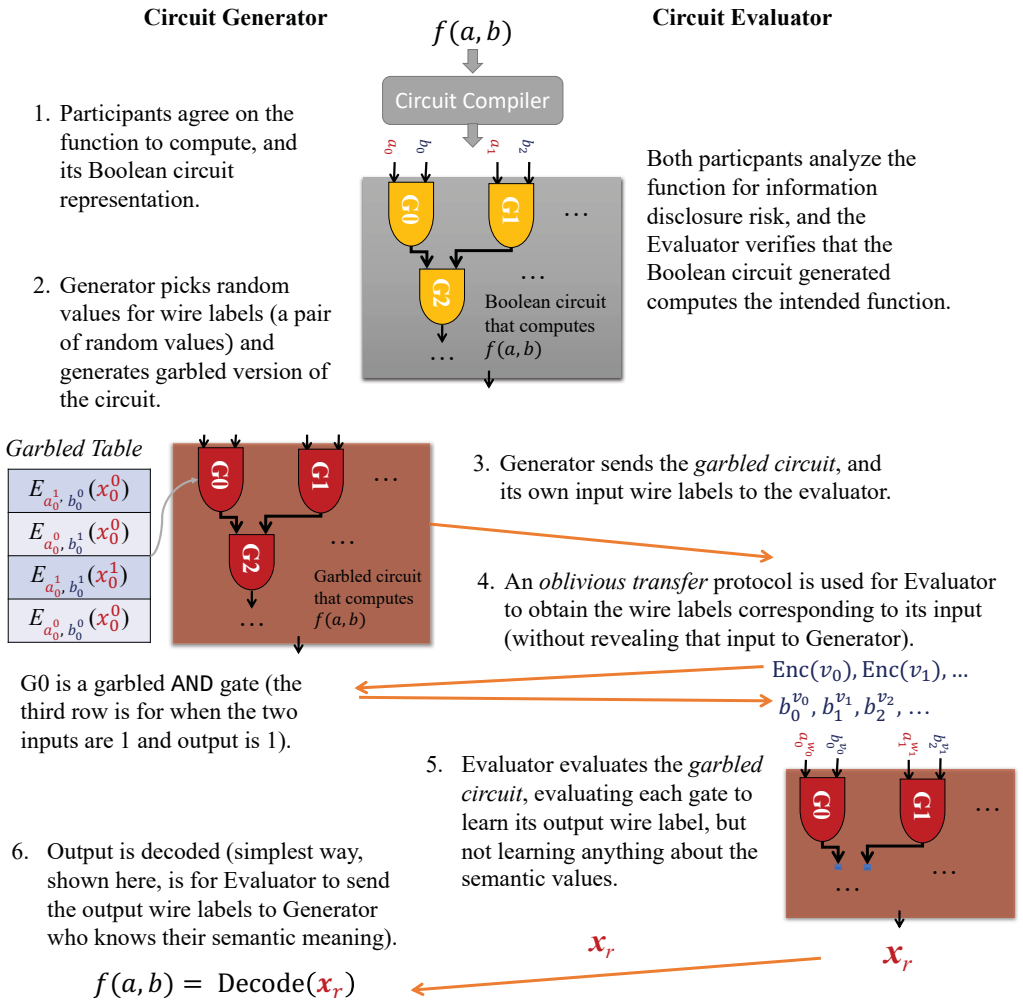


Figure 4-5: Yao's Garbled Circuits Protocol

sponding to the input wire labels it has for each gate during the computation. So, for a given wire w_i, the evaluator can only ever learn either w_i^0 (representing 0) or w_i^1 (representing 1).

Each entry in the garbled table is the wire label representing the corresponding output value, encrypted with the wire labels of the inputs. The key property this has is that the evaluator is only able to evaluate one row of that table, obtaining a wire label that represents its output. Since the evaluator cannot decrypt the other rows in the garbled table, it cannot determine if this output represents 0

(for example, if the gate is an AND and this wire label matches the value in two other rows) or 1 (for an AND gate, the value of a single output row). To enable the evaluator to know which entry to use, one of the wire label bits is used as a "point-and-permute" bit. At the end of the protocol, the final output can be decoded (this could be done either by sending it back to the circuit generator, who knows the keys and can learn the semantic value of the output wire labels), or by adding a decoding step that would allow the circuit evaluator to learn the output.

It remains for the evaluator to obtain the wire labels corresponding to the sensitive input. The generator's input wire labels can just be transmitted directly — the evaluator does not know their semantic value. For the evaluator's input wire labels, the evaluator must obtain the wire labels $w_i^{b_i}$, where b_i is the (semantic) value of the i-th bit of the evaluator's input, without revealing the b_i values to the generator or learning the complement wire labels, $w_i^{1-b_i}$. This requires asymmetric cryptography, and is accomplished through an *oblivious transfer* (OT) protocol. OT-extension protocols have been devised that enable very efficient oblivious transfer for a large number of input bits, using only a number of asymmetric operations that scale with the security parameter and the number of hashes required only scales with the input size) [40].

Although the garbled circuits protocol seems expensive because of the encryption operation needed for each gate evaluation, protocols based on Yao's construction can be surprisingly efficient. Many improvements have been made to the efficiency of garbled circuits protocols [26], and tools have been developed to enable programs written in high-level language to be automatically compiled into garbled circuit executions. Symmetric encryption is remarkably efficient with the built-in AES instructions provided by modern processors, and a series of optimizations to the basic protocol have substantially reduced the number of encryptions and ciphertexts needed including the free-XOR technique [45], which enables XOR operations to be performed without any encryption or cipher-

texts; half gates garbling [98], which reduces the number of ciphertexts needed for an AND gate to two, and hybrid techniques which incorporate oblivious random access memory [32] into MPC protocols to provide sublinear memory access costs [60, 33, 23].

4.3.2 Trust models for MPC

As discussed in Section 4.1, MPC protocols can be designed for a variety of threat models that capture assumptions about adversary capabilities and goals. In addition, MPC protocols can be deployed with a variety of trust models that reflect assumptions about the data owners in an MPC and whom they trust. The cryptographic protections provided by MPC allow each data owner to maintain control over their own data and only rely on their own hardware and software to correctly implement the MPC protocol. In many deployments, however, it is not practical for all data owners to be involved in performing the computation. In other settings, MPC can be used to improve security even when there is only one data owner. To clarify these settings and ways of using MPC, JASON identified three different trust models for MPC deployments; they are described below. Although there are no established names for these models in the literature, each of the models is widely used. Section 4.3.3 includes example applications of each of these models.

Single Steward MPC In the *Single Steward* model, MPC is used to provide additional security for a setting where a single data owner already has all the data required to perform the computation. This model is useful for mitigating insider attacks, limiting the risks of exposure through accidental error or malicious compromise, and increasing confidence in auditing mechanisms. The data are split using a secret-sharing method into two or more shares, each of which are sent to a different server. The servers then jointly perform an MPC to compute the desired function on the data. This setting does not provide enhanced privacy since the

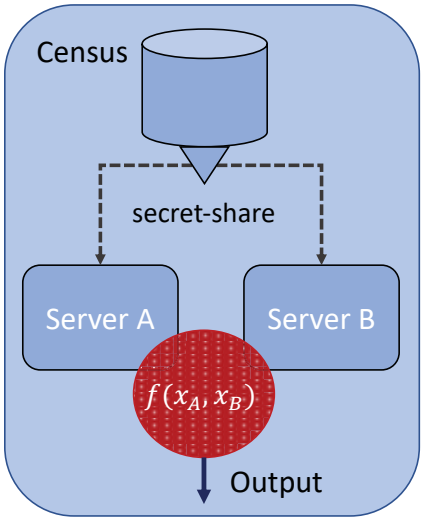


Figure 4-6: Trust model for Single Steward MPC

original data owner, responsible for stewarding the data and managing the process, already has access to all of the data, but, it can still be useful for mitigating the risks of vulnerabilities being exploited by external adversaries. For example, in the setting shown in Figure 4-6, if an adversary can compromise Server A, but not Server B, this protects the data from disclosure. This type of MPC can also be used for deterring insider attacks if the two servers are isolated organizationally, for example, if the systems are administered in a way that no system administrator has access to both servers, so different individuals are responsible for each server. It can also be useful for satisfying legal requirements, such as providing an audit trail, that could be enforced in this model since any function run on the data would require approval by two managers, one responsible for each server.

As depicted, the data is split into two shares and computation is performed using a two-party MPC protocol. Data could be split into more shares, and in cases where three servers are used, there are very efficient MPC protocols for three-party honest-majority settings. In an *honest-majority* protocol, instead of providing security guarantees when any number of parties are dishonest, the protocol only provides security guarantees when the majority of participants behave

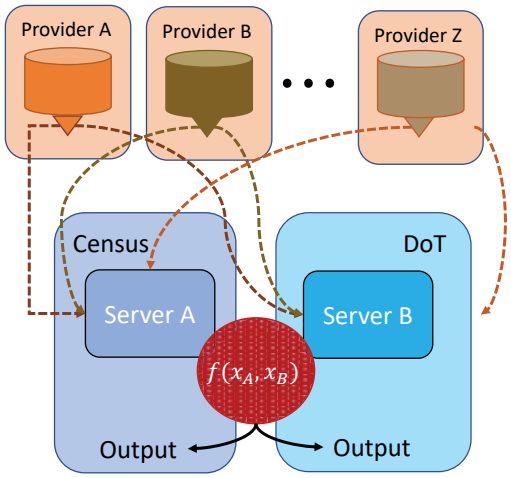


Figure 4-7: Trust model for delegated MPC. In this example, the data providers delegate trust to two server operators, who each obtain a share of the data. The server operators are trusted to not collude, but neither operator by itself can learn the input data. Together, the two servers execute an MPC computation to compute a function on the secret-shared data.

honestly. For a three-party protocol, this means that at most one of the servers is compromised. For many settings, this is a reasonable assumption, and the efficiency advantages of three-party honest-majority protocols outweighs the additional security assumption and the costs of operating a third server.

Delegated MPC In the *Delegated* model, depicted in Figure 4-7, a computation involving sensitive data from many data providers is done without requiring the providers to participate directly in the MPC protocol execution. The data providers still obtain most of the protections offered by MPC, but instead of only trusting their own software and hardware that participate in the protocol, they split their data among two or more delegated MPC servers. Those servers then perform the computation on the secret-shared data using an MPC protocol, and obtain the output of the function. The data providers need to trust the operators of the delegated servers to not collude. If the server operators do collude, they can just combine the secret shares directly and learn the data providers' sensitive input.

If the delegated server operators can be trusted not to collude, however, then the protocol provides protection against any one of the server operators being malicious. Since the delegate servers are only receiving secret shares, which have no semantic value in and of themselves, and then participating in an MPC protocol to perform the computation, they do not learn anything about the other secret shares during the protocol execution. They do, however, learn the output of the function, so it is essential that the data providers trust that the delegated servers will only compute appropriate functions on the provided data. Since both parties in the MPC must agree on the function to be computed, this assurance relies on the same no-collusion requirement as the rest of the data protection. For a data provider to trust a delegated MPC with its secret-shared data, it must be convinced that at least one of the server operators will behave honestly.

As with Single Steward MPC, Delegated MPC can be done with more than two parties, and the most efficient solutions are often in the three-party, honest-majority setting. Note here, however, that the honest-majority setting means that now a data provider must be confident that the majority of the server operators are honest. In a three-party honest-majority delegated MPC, if any two of the delegates collude, all security is compromised; the data providers rely on at least two of the three server operators being uncorrupted.

Joint Data Provider MPC The *Joint Data Provider* model, depicted in Figure 4-8, is the traditional setting for MPC, in which two or more mutually distrusting parties jointly perform a computation on their combined data. In this setting, a participant only needs to trust that the protocol satisfies the desired security properties (which can be proven mathematically for many MPC protocols), and that their own computing infrastructure implements the protocol correctly and does not otherwise expose their data. This is the cleanest model for academic analysis of protocols, since a data owner who participates in a Joint Data Provider MPC

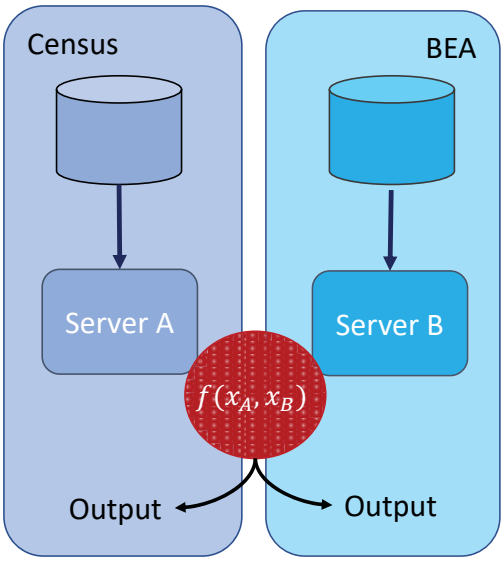


Figure 4-8: Joint Data Provider MPC. Each data owner operates its on server, and only has to trust the server it operates and the MPC protocol.

protocol does not need to trust anything outside of their own control other than the mathematics used to establish the protocol security proof. In practice, however, trusting one's own computing infrastructure is difficult, and nearly always ends up requiring trust in third parties who provide protocol implementations, not to mention, the operating system and processor that executes those implementations. Hence, even in this setting where a provably secure protocol is used, data stewards need to consider the risks stemming from their own computing infrastructure.

4.3.3 Applications of MPC

In this section, we survey some applications of MPC to give an idea of what is feasible today, and aspects that need to be considered in deploying MPC applications. Table 4-2 provides a summary. These examples provide an understanding of the types of applications that can be built using known MPC techniques often in combination with other techniques, how well they scale, and the threat models that can be achieved.

Table 4-2: Summary of Representative MPC Applications. The first four applications are in the passive (semi-honest) threat model; the bottom two provide active (fully malicious) security.

Application	Computation	Model	Cost/Scale
Boston Workforce Study [49]	sum	Delegated 2PC	∼70 employers
Education/Income Study [9]	aggregate statistics with join	Delegated 3PC	10M tax linked with 0.6M education records
Email Spam Filtering [35]	Naïve Bayes classifier	Joint Data 2PC	5× standard cost
Private Set Intersection [62]	private set intersection	Joint Data 2PC	1M items in 4 seconds
Advertising Measurement [46]	private join and sum	Joint Data 2PC	Google user data with advertisers' sales data
Key Splitting [27]	authentication, TLS	Steward 3PC	1B gates/second

Boston Women's Workforce Study In 2013, Boston mayor Thomas Menino pledged to make Boston a destination of choice for working women. The Boston Women's Workforce Council (BWWC) was established to fulfill this aim. One goal of this group was to understand the root causes of the existing wage gap between men and women engaged in similar occupations. To study this, and also to measure ongoing progress in reducing that gap, the BWWC partnered with researchers at Boston University (BU) to develop a secure method for collecting and analyzing salary data. Such information is sensitive for the companies asked to provide it, since they do not want their salary structure exposed. However, gross statistics computed over the population of participating companies in Boston would not be sensitive provided many companies participated in contributing to the survey.

To achieve a privacy-preserving solution, computer scientists at BU developed an MPC solution [47, 49], illustrated in Figure 4-9, and provided a web-based interface for companies to provide their data. A key aspect of their solution is that it only requires a single server operator, in this case the Hariri Institute at Boston University, but performs the computation and generates results in a way

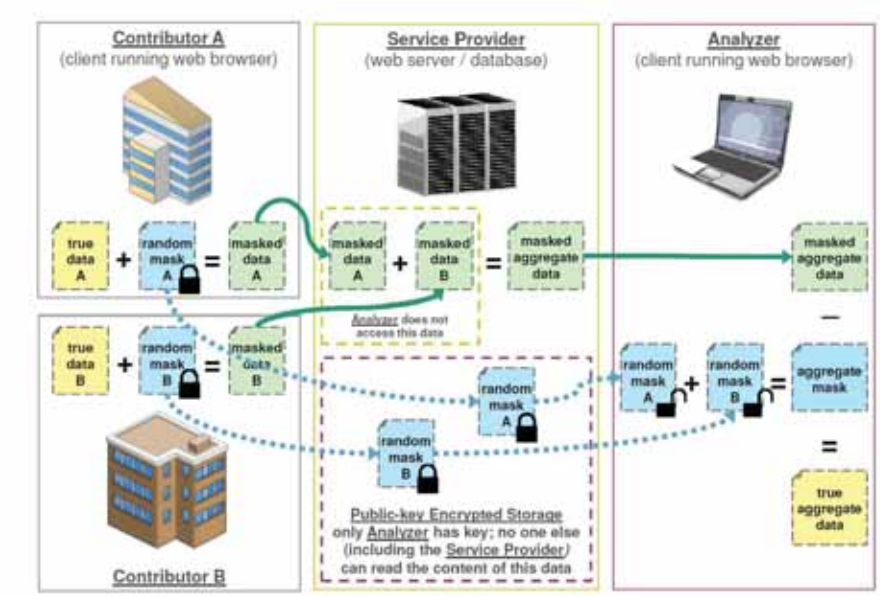


Figure 4-9: MPC protocol used for the BWWC study. Graphic is from [48]

that only a trusted party (in this case the BWWC) can obtain the results. This is done by masking all values with randomly generated keys, where the mask values are encrypted using the public key for the BWWC.

The computation proceeds in four steps:

1. A participant in the study uses a web-based interface to submit the relevant salary data in the form of a spreadsheet as shown in Figure 4-10. The rows of the spreadsheet are labeled according to job description (e.g., executive, technician, etc.). The columns are labeled by ethnicity and also broken out by gender. The web server is located at BU, so skeptical participants would need to audit the scripts provided in the web page before entering any sensitive data into it.

2. Scripts on the web form mask the provided salary numbers by adding a random mask to each entry, and then submit the masked values to a database server also at BU.

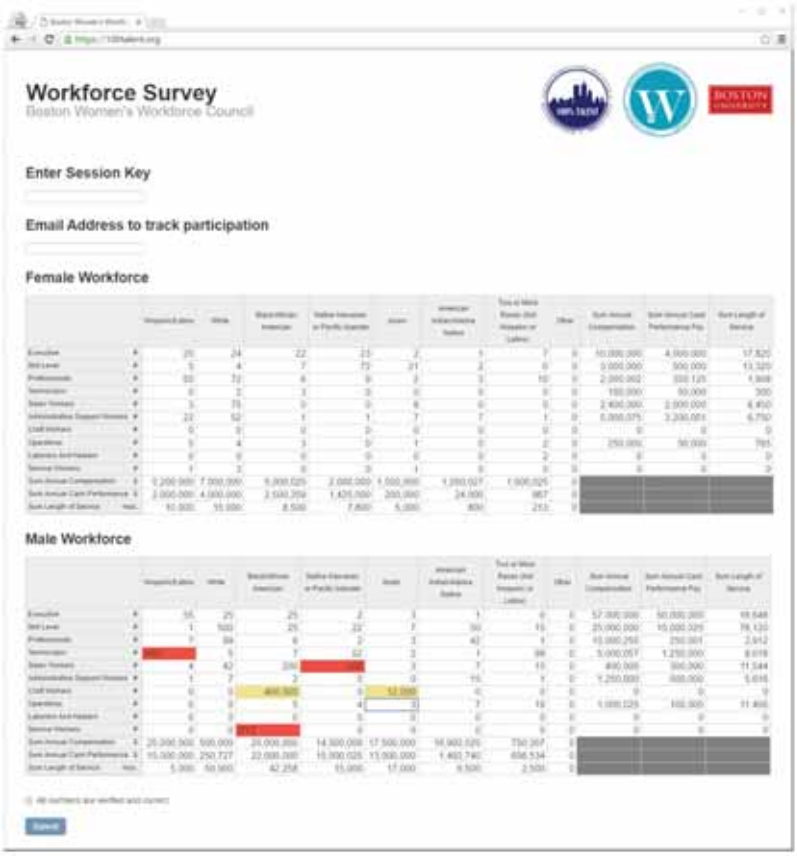


Figure 4-10: An example of the web-based interface used in the BWWC survey. Note that the web-based interface attempts to enforce some data input constraints to prevent erroneous entries. Graphic is from [48].

3. The random mask for each entry is encrypted using a public key provided by the BWWC and sent to a server at BU. These random masks can only be decrypted by the trusted party (in this case the BWWC).

4. The BU server computes the sum of the masked data values, and sends this masked aggregate sum to the BWWC along with the unmodified encrypted masks. The BWWC uses its private key to decrypt the masks and subtracts this from the masked aggregate sum to obtain the result.

This process can be performed on various different data elements to obtain sums broken out by gender, ethnicity and job description.

Note that throughout this process, no party learns any sensitive individual salary data. The approach here works because of additive secret sharing. The masks are a type of additively homomorphic encryption system, so the masked sum can be computed on the masked data, and then produces the correct output when the masks are removed. Note also that the BU server can change the aggregate sum arbitrarily (by adding or subtracting any value to the masked aggregate sum), however, so this only provides semi-honest security.

An important feature of the BWWC survey was the development of the web-based interface. It was considered essential for usability that data could be provided using a simple web form. Since there is no opportunity for data cleaning with the masked values, the web interface was designed to perform consistency and sanity checks locally. For example, checks on plausible salary levels were done by the web form to attempt to prevent errant or nonsensical entries in the spreadsheet.

Estonian Education/Income Study In 2015, statisticians in Estonia wanted to understand the impact of education on income, and, in particular, to see if there was a correlation between students working during their studies and graduation rates. The inputs needed for the study involved income data held by the Estonian Tax and Customs Board, and education data held by the Ministry of Education and Research. To comply with Estonian data protection regulations, the two data sets could not be combined or released. To enable the study an MPC was implemented that allowed computation of joint statistics on the data sets without exposing the data inputs [9].

The MPC was performed using a three-party honest-majority MPC software framework, Sharemind [8] and used the Delegated MPC model. The three servers were operated by Cybernetica (the commercial developer of the Sharemind framework), the Estonian Information System's Authority, and the Ministry of Finance.

Hence, instead of relying on their own computing infrastructure as would have been possible in a two-party Joint Data Provider model involving the data holders, the data holders trusted the three organizations running delegate MPC servers to not collude and to carry out the protocol as specified. The data were protected using secret sharing across the three servers operated by the two government agencies and Cybernetica. Note that all were running software provided by Cybernetica, so security depends on the agencies auditing and trusting that software.

By using the honest-majority three-party model, the computation could scale to the size required, with over 10 million tax records joined with over 600,000 education records. The total execution time for the protocol was 384 hours, which was dominated by the time to aggregate the tax data and compile the analysis table within the MPC. This work was performed in 2015; improvements to protocols and computing power since then make it plausible to infer that a similar computation could be done today, using the same approach, in about 5 hours of compute time.

Private Set Intersection Many interesting data analyses can be viewed as computing an intersection of two or more data sources, and either revealing the result of the intersection or computing some simple aggregate statistics on the set intersection. When the two data sets that are input are kept confidential, this is known as *private set intersection* (PSI). PSI is required for example when one wishes to perform exact record linkage on diverse data sets. Although private set intersection has been implemented using general-purpose MPC [38], very efficient custom protocols have been developed for PSI that take advantage of the fact that its main operation is an oblivious equality test. An example of a recent PSI protocol from Pinkas et al. [62] uses an oblivious Bloom filter construction [2]. Both the communication and computation costs scale linearly in the size of the input, and the

[2] A Bloom filter is a data structure designed to determine in a rapid and memory-efficient way, whether an element is present in a set.

protocol can perform a PSI on sets with one million items with fully-malicious security in the Joint Data Provider MPC model in 4 seconds on commodity processors. Experiments have not been done on larger sets, but it seems reasonable to expect intersections of billion-item sets could be computed within a few hours. An example deployed application using private set intersection has been demonstrated by Google and its advertisers to allow measurement of the effectiveness of advertising on purchases [46, 39]. This involves computing the private set intersection between advertising click data owned by Google, linked with identities from transaction data owned by the advertisers, to count the number of purchases correlated with advertising. This work also employed a highly tuned algorithm to compute the PSI.

Key Management The single data steward MPC model is useful when a single organization holds all the data for the computation, but wants to protect it from vulnerability to a single compromise which could be either an external attack on a server, or an insider attack. An industrial example is the "virtual Hardware Security Module" (vHSM) application from Unbound Tech [74]. This used a three-party MPC to allow an organization to split its secret keys across three hosts, while enabling operations such as signing that require the use of those keys to be performed as an MPC protocol. Such a model can provide high performance using optimizations that take advantage of specific properties of the application, as well as the three-party MPC setting, executing over 1B gates per second even under an active security model. This performance is sufficient to implement cryptographic operations with reasonable latency,e.g., performing RSA key generation in around 30 seconds) [27].

Other applications Academic researchers have demonstrated MPC as used in numerous applications including detecting satellite collisions [37], implementing the national medical residency stable matching algorithm [22], financial risk anal-

ysis [1], and genomic analyses [15, 96]. Here, we provide some more detail on one example application (included in Table 4-2). Gupta et al. [35] developed a privacy-preserving system for e-mail classification that allowed end-to-end encryption of e-mails, so they would not be visible to the email service provider, while allowing the e-mail provider to perform spam filtering and topic extraction. This was done using a two-party Joint Data Provider MPC where one party is the e-mail recipient and the other party is the e-mail provider. The e-mail sender is involved in that they encrypt the e-mail to the receiver using a standard end-to-end encryption scheme such as OpenPGP, but otherwise is not involved in the MPC. The system than can perform spam filtering using a Naïve Bayes classifier as an MPC implemented using Yao's garbled circuits protocol that protects the e-mail contents from the provider, and protects the spam classification model from the recipient. Since all the operations needed for the classification are linear, this can be performed efficiently within the MPC (about 358ms per e-mail with a large (5M features) classifier model). Many other privacy-preserving machine learning applications have been built using MPC, including training neural networks [99, 55], neural network inference [64, 42, 52], linear regression [58, 28], and evaluating decision trees [97].

4.4 Trusted Execution Environments

Another approach to secure computation relies on placing trust in an external computing environment out of the data owners control, known as a *trusted execution environment* (TEE). The common approach to implementing a trusted execution environment is to use a *secure hardware enclave*, where a hardware vendor is trusted to produce a processor that implements an execution environment which

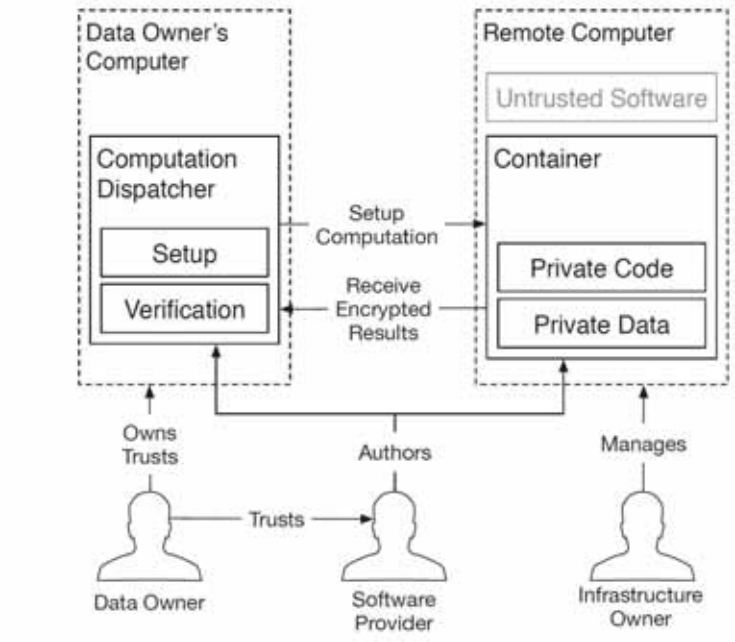


Figure 4-11: In secure remote computation a user relies on a remote server that provides assurances of confidentiality and integrity to perform a sensitive computation. Graphic is from [18].

can be trusted even when operating on a host that is otherwise not trusted. The idea, illustrated in Figure 4-11, is that a user can execute potentially sensitive software on an untrusted server if that server can provide some facilities to guarantee confidentiality and integrity.

CPU manufacturers have attempted to provide TEEs by combining special instructions, encrypted memory, and some way of assuring users they are using the TEE. Intel's Software Guard Extensions (SGX) is a primary example, and it is implemented by Intel processors intended for both consumer and server machines. The goal is to let an external user send code and data to the secure enclave over an encrypted channel, run the code on the data in the enclave, and return the results over the encrypted channel. An observer in the cloud, even a host operating system on the same CPU, should not get any useful information about the computation or the data.

As will be further discussed below, there are reasons to be skeptical about this endeavor. In general, almost all complex software systems are buggy in ways that affect security. SGX in particular has failed to live up to its security guarantees since it was released. Beyond the usual challenges, however, the inherent design goal of SGX raises additional reasons to be doubtful that it can ever achieve its stated aims. The goal of SGX is to allow a process protected in an enclave to share resources with a general purpose processor that is also executing untrusted programs, and the goal of sharing resources without leaking information poses considerable challenges. It is known to be remarkably hard to keep close observers from learning useful information about a running program. Put the other way, it is remarkably hard to keep a program from leaking useful information even when the available observation channels are narrow and well confined. For hardware enclaves like SGX, the challenge is exacerbated by the amount of hidden state in modern CPUs, particularly various caches, that are shared among processes. The state of these resources is architecturally invisible (that is, not readable by provided instructions), but can be deduced through indirect means whereby an observing process can learn information about another process sharing the same CPU. Spectre [43] and Meltdown [51] are two well-known examples of attacks that exploit this shared microarchitectural state; the original attacks did not compromise SGX, but subsequent similar attacks did, as discussed in Section 4.4.2. Spectre exploits a branch prediction cache; Meltdown exploits how speculative execution, a crucial performance feature, affects the memory cache.

4.4.1 Overview of Intel SGX

In this section we discuss aspects of Intel's SGX approach to providing a trusted execution environment. The idea is to leverage trusted hardware on a remote computer to provide both confidentiality and integrity of a program that performs a potentially sensitive computation. A number of trust relationships are required

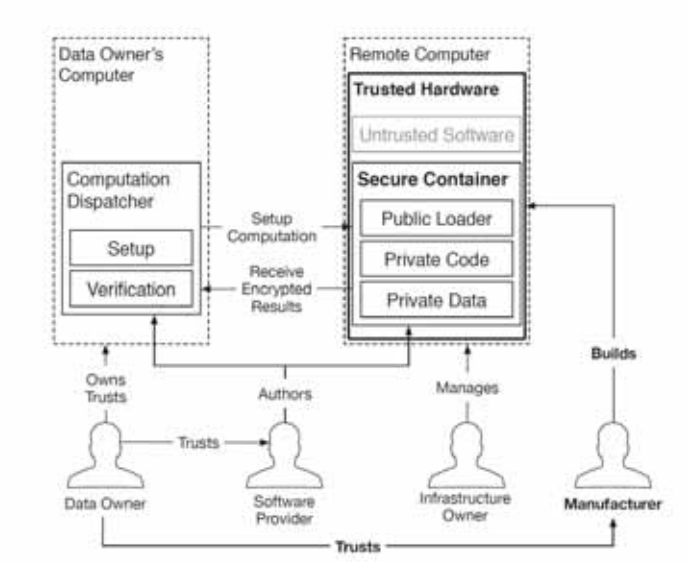


Figure 4-12: The trust relationships associated with secure enclaves. Graphic is from [18].

to accomplish this as shown in Figure 4-12. The data provider must trust the developer of the software and so the nature of the queries being made by the software must be checked for potential disclosure. Beyond this, the data provider must also have confidence that the designer of the SGX enclave does not provide inadvertent paths for data leakage.

The trusted hardware provides a sealed environment which in principle cannot be accessed even by a corrupted host operating system. There are several steps in running code under SGX:

1. A user must first verify that they are communicating with software hosted by the secure enclave on the remote server tasked with performing the computation. This is accomplished using software attestation in which a cryptographic key exchange provides a hash of the contents of the enclave. This establishes that only the remote user's data and software can be loaded into the enclave as the computation proceeds.

2. The remote user must also verify the attestation key used to sign the attestation. This is accomplished by checking against an endorsement certificate provided by an external remote Intel attestation service. This certifies that the remote service is running a verified Intel SGX enclave. This endorsement is done blindly, so that the specific processor enclave is not revealed to Intel.

3. The remote user can then send private data requiring secure processing over an encrypted channel to the enclave. The data are then processed within the enclave, using the software that was attested to in the first step. Results can then be returned to the remote user over the encrypted channel.

At the end of a successful attestation process, the external user will have confirmed that it is communicating with a value enclave containing the expected software, and will have a symmetric key to use to establish a secure channel to that enclave.

In order to provide the desired security properties, SGX enclaves set aside in memory a protected region called the Processor Reserved Memory (PRM). The CPU hardware prevents any access to this memory from non-enclave sources. This includes even privileged software like the operating system kernel, the hypervisor (software that manages virtualization), and even hardware controls built in to the server motherboard such as the System Management Monitor. The PRM is composed of 4KB encrypted pages of memory containing the private data and code. The total amount of such memory is currently 128 MB, but future SGX implementations may provide additional protected memory.

For applications that require more memory, data can be stored in main memory and one of the features of SGX is automatically encrypting memory from the enclave before it is moved to main memory, and decrypting the encrypted data when it is reloaded into the enclave. The pages associated with a given enclave

are accessed through an enclave page cache (EPC). The CPU enforces the association between a given page and its associated enclave. When protected code and data are to be executed by the CPU they are decrypted rapidly by a hardware memory encryption engine. The computations themselves are performed in the clear by the CPU functional units and the results are then re-encrypted for storage in the PRM. When main memory is used, however, it is important to ensure that the access patterns (which are visible to the host and processes outside the enclave) do not reveal sensitive data. This can be done using Oblivious RAM (ORAM) techniques [67], but for many applications this induces a significant performance penalty. It also only hides which memory block is being accessed, but not the timing and number of memory accesses, so ORAM by itself may be insufficient to prevent inferences from revealed access patterns.

During the initialization process code and data that are not viewed as sensitive are loaded from unprotected memory into the enclave. Once this loading process is completed a cryptographic hash of the contents is computed as a way of verifying to the remote user the initial contents of the enclave. The attestation process is used to assure a user that they are communicating with a valid SGX enclave that contains the code as loaded by the user.

The execution of code in the enclave is achieved using a special set of instructions. This is similar to the context switch that takes place when kernel level code is executed but the execution in the enclave itself occurs at the lowest level of privilege for the processor. In fact, processes at a higher level of privilege such as the kernel cannot execute enclave code. This is meant to protect the secured code from a possibly corrupted kernel, but is also meant to protect the operating system from malicious or errant enclave software. If an interrupt is signaled or there is a page fault or other exception, the CPU does not directly service the fault, exception, or interrupt as it would in normal operation. Instead, the CPU first saves the enclave state in memory thus ensuring it is encrypted, and then transfers con-

trol outside the enclave. In this case, the CPU does overwrite registers to prevent leakage of information

Since SGX runs programs on the decrypted data, programs running in the enclave can have similar performance to normal executions with unencrypted data on the CPU. There is some overhead because of the limited memory and instructions available within the secure enclave, and because of the need to encrypt data before it is stored in main memory (and to decrypt when data is restored). This means many applications can run within SGX with overheads of 10-20% over runtimes in a conventional CPU. This assessment, however is heavily dependent on the type of application, since to execute securely within SGX requires carefully designing the program to not have any memory access pattern or timing side channels that would be observable outside the enclave. For many programs, the changes needed to make them side channel free are complex and substantially increase the running time.

4.4.2 Security issues

The development of SGX was a notable step forward in enabling widespread trusted computing, and major cloud providers are offering secure computing services using SGX and similar competing technologies from other vendors. However, as we describe in this section, SGX has suffered repeated security issues since its inception. We cover here only some of the issues with the purpose of indicating the nature of the difficulties that have been uncovered.

All of the attacks described here assume that the operating system has been compromised and that the attacker has basically full control of the entire computational environment. This may seem like an extreme assumption, but it is exactly the threat model for which SGX is designed. In addition, there have been many demonstrations that once an attacker has identified a vulnerability in a system, it

can often be amplified to achieve a full host compromise. Recall that the security guarantees claimed by SGX provide integrity and confidentiality even in the presence of a compromised host.

Foreshadow Modern processors use a variety of techniques to hide the latency between the execution of instructions in the CPU and access to data in memory. For example, memory caches were developed to speed up delivery of data to the processors. In addition, most modern processors use techniques such as speculative execution to predict the value of various operands that are being fetched from memory and thus continue executing instructions. When the memory access completes, and the actual instruction to execute is determined and fails to match the speculatively read operand, the system is able to roll back the speculative execution and no harm is done to the (architecturally visible) state of the program. If, however, the prediction was correct, the CPU has effectively hidden the latency incurred by the memory access.

In 2018, security researchers [51] revealed that speculative execution in Intel processors could be used to break the memory isolation between the kernel and user memory spaces, thus allowing leakage of secret data. This was accomplished by causing an errant read of data in a protected area of memory. When this occurs the processor throws an exception and invokes a handler to respond to the fault. However, the processor continues to execute speculatively. This gives an attacker time to manipulate the processor cache and thus leak sensitive kernel data.

This attack in and of itself cannot be used to leak secret data from an SGX enclave because, as mentioned previously, SGX enclaves do not use exceptions when illegal accesses of enclave memory occur. However, SGX does use standard exception handling for memory management. This is required because the operating system must be able to swap memory pages in and out. Since the attacker has complete control of the system, they can mark SGX memory as unavailable

and trigger an exception and the concomitant speculative execution as the exception is handled by the operating system. Since SGX data is not encrypted when it enters the cache, the secret information can be extracted [93]. In fact this problem affects not only SGX, but breaks isolation of virtual machines running under a hypervisor. This vulnerability was given the name Foreshadow to highlight the use of speculative execution.

CacheOut After the demonstration of the Foreshadow exploit, Intel mitigated the issue by modifying the hardware in the latest Core i9 series.[3] However, security researchers demonstrated that it was still possible to leak sensitive information though a set of mostly undocumented buffers used by the CPU to load data in and out of the cache. This type of attack by itself, known as Microarchitectural Data Sampling (MDS), is not terribly practical; it has been compared to "drinking from a fire hose" because it was not thought to be possible to control what data were transiting through the various buffers. In addition, to counteract MDS-type exploits, Intel re-enabled a legacy instruction to clear the various buffers so that they could not be read as a result of a fault-type attack like those described in the section above on Foreshadow.

Intel's buffer overwrite mitigations, however, are not completely effective. The various buffers are also used when the cache evicts data and the operation is non-blocking. In the CacheOut attack [95], an attacker first evicts data from the Level 1 cache, and subsequently uses a faulting or assisting load to recover it from one of the undocumented buffers. This is even effective even on Intel's latest generation of processors that have hardware countermeasures against exploits like Foreshadow and MDS. Security researchers have demonstrated that this type of manipulation of the Level 1 cache can leak a variety of sensitive information including data from SGX enclaves. Schaik et al. [95] demonstrated using this at-

[3] Software mitigations were developed for processors manufactured prior to 2019, but because these mitigations are performed in software, they do incur a performance penalty.

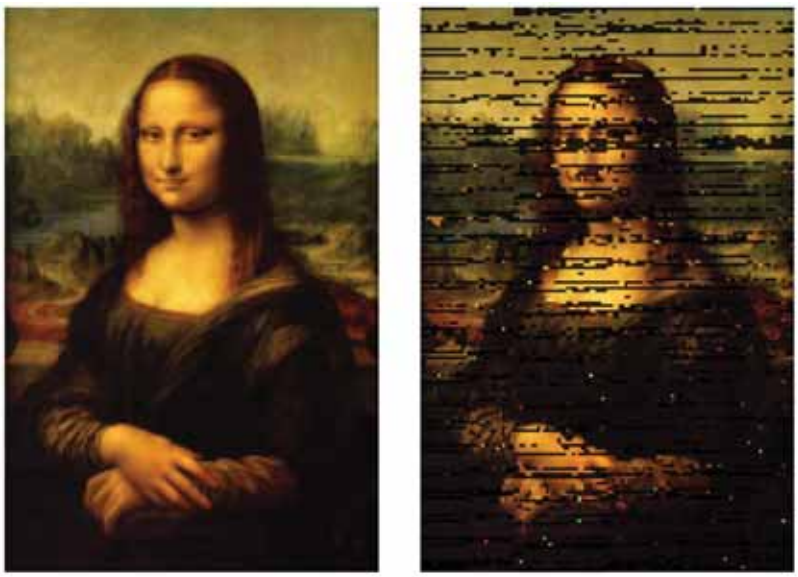


Figure 4-13: Extraction of an image from an SGX enclave using CacheOut. Graphic is from [95]

.

tack to extract image data stored in an enclave as shown in Figure 4-13. Because the adversary does not have total control over the operation of the processor, the recovery of the image is not perfect but it is certainly recognizable. This attack does require control of the host operating system, but it again violates the security guarantees of SGX on Intel's most advanced hardware.

SGAxe [94] The CacheOut exploit described above was used to extract a decrypted Enhanced Privacy ID (EPID) key used for attestation of SGX enclaves. The Intel attestation service provides verification that the enclave to which the user is communicating is a genuine SGX enclave by validating an attestation key derived from keys stored in the hardware. These EPID keys are generated by a quoting enclave and sent to the remote user. EPID keys are never meant to appear in the open. Instead they are used as part of a key exchange protocol to generate a session key that secures a private channel between the user and the enclave. Moreover, these keys are unlinkable, in that the key verifies an enclave is genuine, but

does not provide information about the processor associated with the key. This blind attestation property is a feature of the attestation protocol, which was designed to ensure that enclave users maintain some privacy rather than needing to disclose to Intel every time a particular enclave is used. This property means, however, that an enclave user is not verifying a specific enclave, but only that the enclave is valid and contains the expected data in memory.

A leaked EPID key can be used to forge attestations that convince a remote user that they are communicating with an authenticated enclave when, in fact, they might be communicating with software simulating an enclave. This, in principle, constitutes a breakage of the entire SGX attestation ecosystem.

Plundervolt Modern commercial CPUs provide special purpose software and hardware interfaces to control processor voltage and frequency. These interfaces are privileged, but could be used by an attacker that has arbitrary access to the operating system. The interfaces are made available because some users wish to overclock processors to increase performance while others wish to dissipate less power from a processor when it is idle. However, the processor is not guaranteed to return correct results for all possible settings of voltage and frequency that can be accessed using these interfaces. While this can produce computational faults, it can also be used to violate SGX security guarantees. Security researchers were able to show, for example, that they could use these interfaces to impose faults in the middle of an AES encryption and thus infer a confidential AES key stored in SGX. More concerning is the ability to have SGX write data stored in encrypted memory in an unencrypted form to memory outside the enclave. This exploit was named Plundervolt [56]. Intel has released several firmware patches that mitigate such attacks. The patches lock the voltage settings on processors and must be removed if one wishes to modify the voltage or frequency. It is thought that executing an attack using this approach is most likely quite difficult

but it does underscore the fact that the software and hardware environment of a modern processor is sufficiently complex that guaranteeing security is a significant challenge.

Attempted Mitigations As new attacks on SGX are demonstrated, Intel has released patches aiming to mitigate attacks. Attacks done by responsible security researchers (including all the examples mentioned earlier in this section) are reported to the vendor through a responsible disclosure process, and kept from public release until the vendor and users have an opportunity to mitigate the disclosed vulnerability. Hence, most of the discovered attacks were not reported until after mitigations could be developed.

Figure 4-14, from Schaik et al. [95], shows the status of various generations of Intel processors relative to the vulnerabilities discussed above. As can be seen in the Figure, while some of the vulnerabilities discussed above have been mitigated, exploits like CacheOut have not. Since the hardware cannot be modified after production, many of the mitigations require modification of the processor microcode or operating system software. This can incur performance overhead. The hardware mitigations, designed into subsequent generations of the processor, may also incur performance cost if they require flushing shared state or restricting the use of hardware resources.

The point of this section is not to disparage the objectives or design of Intel SGX. Indeed, one gets an appreciation for the complexity of modern processors and the ingenuity of Intel's engineers in designing and adding a TEE to a complex, performance-driven processor. The main conclusion is that, given the complexity and amount of shared state in modern processors, it is unlikely that all leakage issues can ever be mitigated with this design.

CPU	Year	CPUID	Meltdown	Foreshadow	MDS	TAA	CacheOut
Intel Xeon Silver 4214 (Cascade Lake SP)	Q2 '19	50657	✓	✓	✓	✗	✗
Intel Core i7-8665U (Whiskey Lake)	Q2 '19	806EC	✓	✓	✓	✗	✗
Intel Core i9-9900K (Coffee Lake Refresh - Stepping 13)	Q4 '18	906ED	✓	✓	✓	✗	✗
Intel Core i9-9900K (Coffee Lake Refresh - Stepping 12)	Q4 '18	906EC	✓	✓	✗	✗	✗
Intel Core i7-8700K (Coffee Lake)	Q4 '17	906EA	✗	✗	✗	✗	✗
Intel Core i7-7700K (Kaby Lake)	Q1 '17	906E9	✗	✗	✗	✗	✗
Intel Core i7-7800X (Skylake X)	Q2 '17	50654	✗	✗	✗	✗	✗
Intel Core i7-6700K (Skylake)	Q3 '15	506E3	✗	✗	✗	✗	✗
Intel Core i7-6820HQ (Skylake)	Q3 '15	506E3	✗	✗	✗	✗	✗

Figure 4-14: The status of various Intel processors vulnerability to vulnerabilities discussed in Section 4.4.2. A green check-mark indicates the vulnerability has been mitigated. A red X indicates the processor remains vulnerable. Table is from Schaik et al. [95].

In their presentation to JASON [16], Jonan Chu and Daniel DeGraaf of the NSA confirm that the speculative execution vulnerabilities and the presence of side channels show that some trust must be placed in platform software and the security of the external computing environment. For example, the Defense Information Security Agency (DISA) requires that in order to process Controlled Unclassified Information (CUI) such as data protected under Title 13 or Title 26, the data center must be located in the US, the administrators must undergo background checks, and there must be physical separation from commercial tenants. Such restrictions seem all the more challenging if one wishes to link multiple enclaves to perform a distributed computation. Further, if such protections can be ensured, it is not clear that the SGX protections add much—they are designed exactly for the opposite scenario, where the surrounding computing environment cannot be trusted at all.

In summary, SGX cannot be relied on to provide bullet-proof security in a hostile environment. That does not mean it is without value, although its potential value for US statistical agencies is unclear. Many security discussions, especially as regards SGX, are phrased as if the only threat is an attacker. Attackers are a threat, but various sorts of failures commonly result in leaks of confidential data. Commercial clouds are complex environments with layers of authentication, authorization, and configuration. Simple web searches often turn up open servers

with confidential data exposed. Even if simple mistakes are avoided, the whole setup is somewhat fragile. Vendors introduce improvements and new tools all the time. On the whole this is progress, and good. But it is a common occurrence that configurations that were adequate in one environment may not be later, and that upgrades that were supposed to be backwards-compatible sometimes are not. A few of these unfortunate circumstances can combine to leave accessible things that were meant to be inaccessible. And it is here that TEEs, and SGX, despite its vulnerabilities, may have some value. SGX provides additional disclosure protection as it is keeping its confidential data in encrypted memory. That is, although SGX does not provide the claimed security properties under its intended threat model, as long as it is not used to provide a false sense of security it can be an additional layer of protection against mistakes.

4.4.3 Future approaches to secure enclaves

As discussed in the previous section, implementing a secure enclave on a production processor is challenging. Modern Intel processors make use of techniques such as speculation and hyperthreading to enhance performance. But despite the presence of an enclave, these features make it difficult, if not impossible, to securely isolate a computation. As has been seen above, features like speculation rely on sharing architectural state and this can be exploited to leak information.

JASON was briefed by Prof. Srini Devadas of MIT who described ongoing efforts to design and build processors where isolation is built into the hardware from the ground up. Devadas described two designs.

The first, called Sanctum [19] makes minimal changes to the hardware of an open source RISC V ISA core. The processor provides multiple cores but does not support speculative execution or hyperthreading as these features require sharing among processes of the micro-architecture. The architecture, which has been

implemented on a field programmable gate array (FPGA) employs the following strategies to enforce isolation:

- data caches are isolated using partitions,
- page tables are isolated inside the enclave,
- cache rows are partitioned using hardware support,
- DRAM pages are colored, so that the partitioning in the cache matches the striping in DRAM,
- information transferred to insecure memory is encrypted, and
- oblivious RAM is used to hide memory access patterns.

These strategies do incur a price in terms of performance, particularly the lack of speculative execution which can significantly boost processor performance. Interestingly, the hardware modifications required to implement the measures above are rather minimal.

Because speculative execution is considered essential for performance, a more advanced processor design called MI-6 [10] is also being developed. In this design, data buffers are cleared every time a processor security monitor switches contexts. In addition a processor core that executes enclave code disables speculation. At present, hyperthreading cannot be supported while still preserving isolation.

At present, such processors that enable security as a "first class citizen" are not likely to be widely available on a five year time horizon, but it remains important to track progress in this area as availability of such processors would make it possible to run securely even in the presence of a compromised host computer. If a sound secure enclave could be implemented that satisfies the goals envisioned

for SGX, it could allow for a self-contained program to be audited for desired disclosure properties, and for external users to know that this is the only program that will have access to their data.

4.5 Programming for Secure Computation

Designing new cryptographic protocols and proving their security requires a high level of expertise and experience. Over the past decade, however, there have been many efforts to make secure computation more accessible to developers without cryptographic expertise. This section provides a brief introduction to what must be done to program applications using secure computing technologies, with a focus on what it would take to turn existing data analysis implementations into efficient and secure computations.

4.5.1 Circuit-based protocols

Both general-purpose FHE and MPC protocols operate on either Boolean or arithmetic circuits. This means algorithms that execute as secure computations must be converted to fixed circuits, where the circuit generated depends on the input size, but otherwise cannot depend on the sensitive data.

This means we cannot have traditional programming structures like **if** statements where the inner block executes only when the tested predicate evaluates to true. In a secure computation where the predicate depends on sensitive data, it is necessary to hide from the evaluator whether or not the predicate was true, and hence, whether or not the inner block executes. Similarly, we cannot have loops where the loop bound depends on sensitive data. If the evaluator can determine how many times the loop body executes, and that number depends on sensitive data, this would leak information about the sensitive data the secure computation is supposed to protect.

Hence, all secure computations must execute in a *data oblivious* way. This means that anything that is observable about the execution cannot depend on sensitive input data. If the computation is represented by a fixed circuit, this property is easily established. But, representing programs as circuits is unnatural and difficult for programmers, and has the further problem that the circuits for most interesting secure computations become too large to store in memory as the input size scales. Recall that unlike hardware circuits where gates can be reused, in a secure computation protocol such as Yao's garbled circuits protocol, each Boolean gate can only be executed once; evaluating the same gate more than once would leak information about its inputs, so each gate that executes during a computation must have its own garbled table of ciphertexts and, as a result, interesting computations require many billions of gates).

To help manage the construction of secure computations while satisfying the programming constraints discussed above, numerous programming frameworks have been developed that allow secure computations to be described in a way that is similar to traditional programming, but that compile those programs into secure computation protocols. Tools exist for both FHE and MPC, with some similar aspects such as converting a program to an circuit that will be efficient to execute. Other aspects are specific to the FHE or MPC protocol that will be used. In addition, many tools provide support for hybrid protocols, where specialized protocols and state can be used to provide oblivious random access memory and other functionalities.

Many tools and libraries have been developed to enable programmers without cryptographic knowledge to compile programs into secure computation protocols. Hastings et al. [36] provides a survey of tools for programming MPC applications. One advantage these tools provide is that although it may require more understanding of the protocols to produce an efficient implementation, there is little risk that a programmer will use these tools in a way that produces an in-

secure implementation—security is ensured by the cryptographic protocol, and as long as the implementation provided by the tool is correct, there is nothing a programmer can do to violate these. The risk is with the outputs that are decrypted at the end of the protocol, so it is still important that a full disclosure analysis is done on the function to be computed to understand what could be inferred from these outputs.

4.5.2 Secure enclaves

Adapting programs for execution in secure enclaves poses the opposite difficulty. Programmers who lack deep understanding of the security issues involved in secure computation will produce efficient implementations with little effort, but it is likely that those implementations will not provide the desired security properties (even if the enclave is invulnerable and provides the promised protections).

The SGX mechanisms provide trusted execution within the enclave, but do not ensure that the externally observable behavior of the program will not leak sensitive data. This externally observable behavior includes the memory access patterns of the program that are revealed whenever storage is off-loaded from the limited enclave storage to main memory. Although the contents are encrypted with a key maintained by the enclave, and so are not visible to the external adversary, the reads and writes to main memory are visible and can potentially reveal sensitive information about the data in the enclave if the memory access pattern depends on that data. Another source of leaks is timing side channels, which can often be exploited to leak cryptographic keys. If the execution time of a process in the secure enclave depends on the sensitive data, an adversary who can observe those timings may be able to infer sensitive data. Programs in SGX enclaves must also avoid other vulnerabilities. For example, it has been shown that memory corruption vulnerabilities in programs running within SGX enclaves can be exploited to leak data from the enclave [17].

The use of SGX or any other TEE does not absolve the programmer of insuring that the execution does not exhibit side-channels that could leak information. Hence, to provide the desired confidentiality properties, programs that execute in SGX enclaves must be carefully written and audited to ensure that nothing observable about their execution depends on sensitive data. There are constant time implementations of many cryptographic primitives available, and methods such as oblivious access to RAM can be used to hide memory access patterns, but these need to be used carefully; their use can incur significant performance overhead for many applications.

This Page Intentionally Left Blank

5 CONCLUSIONS

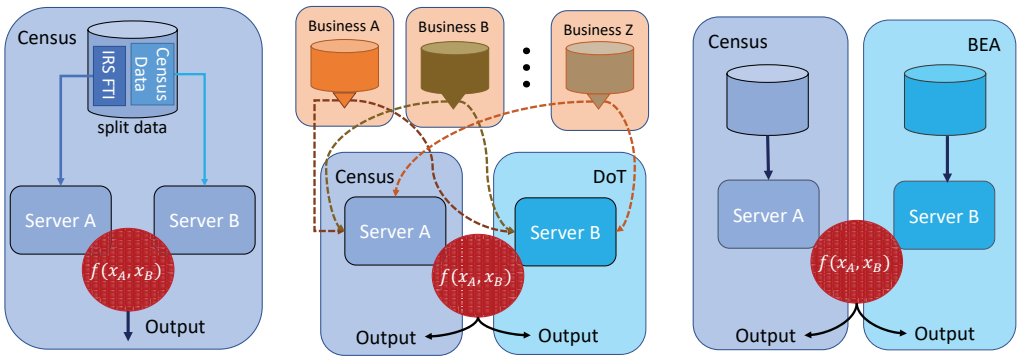
To conclude, this section first suggests specific ways to use MPC for Census Bureau use cases, then provides a general discussion of the importance of thinking about and managing trust in secure computing systems. Section 5.3 gives responses to the sponsor's questions.

5.1 Mapping MPC Technologies to Census Bureau/BEA applications

Having examined the three Census Bureau use cases discussed in Section 3 and briefly surveyed current approaches to secure multiparty computation, we discuss here potential mappings of these technologies to the use cases. We focus here on the use of MPC as JASON believes this provides the best short term opportunities for the required computations. While its capabilities are promising, fully homomorphic encryption is far from being practical for the proposed Census Bureau use cases. It should be noted, however, that aspects of homomorphic encryption (typically additively homomorphic encryption) are used in many MPC protocols. The use of Trusted Execution Environments (TEEs) at present does not appear to offer more security than what could be achieved in a well-managed data center.

5.1.1 Income statistics by business size

This use case (introduced as Use Case 1 in Section 3.2) could be handled via the single-data steward MPC trust model. As the Census Bureau holds all the requisite data, including FTI from IRS, this computation could be performed entirely within the Census Bureau IT environment. However, it presents an opportunity to explore the use of MPC for future applications. In addition, as discussed in Section 4.3.2, the use of the single-data steward model has security benefits even



(a) Income Statistics **(b) Commodity Flow Survey** **(c) Distributed Record Linkage**

Figure 5-1: Mapping MPC technologies to Census Bureau use cases: (a) generating income statistics by business size (Use Case 1); (b) direct processing of business data (Use Case 2) using the Commodity Flow Survey as an example; (c) distributed record linkage (Use Case 3)

within a trusted IT environment like the Census Bureau including providing more protection against insider attacks or mistakes, as well as simplifying auditing since all uses of the protected data must now go through a well-defined interface. The implementation envisioned here is shown graphically in Figure 5-1a. Referring to the Figure, IRS FTI data would be secured on one server (Server A), whereas the required Title 13 Business Register Data would be secured on Server B. A private set intersection protocol would executed within an MPC to match the establishment name and obtain its size. The requisite income fields would then be summed, also within the MPC, to create the required statistics by establishment size which would then be decoded and released. With this design, the FTI data would only be exposed to Server A which could be isolated and carefully managed, and all computations with that data would be performed within an MPC protocol. This would not obviate the need for careful information disclosure analysis of all the functions computed on this data within the MPC to know that the outputs of those computations do not reveal sensitive information about the FTI inputs.

5.1.2 Processing business data

The approach used in the BWWC survey, discussed in Section 4.3.3 has some similarities with the use case (introduced as Use Case 2 in Section 3.3) in which multiple data providers contribute sensitive data to an economic survey. Since data providers could not be expected to operate their own servers to participate in an MPC protocol directly, the delegated MPC model would be appropriate. Each participating business would provide the requisite data using secret sharing across two or perhaps more servers operated by different organizations. One of these could be at the Census Bureau while the others would be owned by trusted intermediaries. Such an intermediary could be the partner agency in this survey, the Department of Transportation (DOT), as shown in Figure 5-1b, but could also be a completely independent entity that is partially trusted by all parties. The trust level needed here is different from that required in the single data steward model—instead of needing to completely trust one organization to steward its data, the businesses providing data would need to trust that the server operators do not collude, and that at least one of the server operators carefully analyzes the functions that are computed on the provided data for information disclosure risks.

One challenge in deploying this solution is that because the data is hidden, there is no way to manually correct for erroneous data. Hence, infrastructure would also have to be established to ensure that input data were consistent, and this requires some development by the Census Bureau to provide a means for the business to upload the relevant data from a company's database system to the Census Bureau. Thought must also be given as to how one would handle missing data either via imputation or simply by exclusion (discussed further in Section 5.1.4).

5.1.3 Distributed record linkage

Distributed record linkage (introduced as Use Case 3 in Section 3.4) is in some ways the most standard use of MPC, as illustrated in Figure 5-1c. In this setting, two or more agencies interact and perform a secure computation, such as a private join and statistical computation on the intersection, while preserving the privacy of their input data. Figure 5-1c shows an interaction where BEA and the Census Bureau perform a joint computation with two participants. Applications like LODES and OntheMap would require an MPC involving more participants such as state tax authorities. There are MPC protocols that can scale well to support many participants, but successful deployments like this become more complicated since all the participants must manage servers that are online and executing compatible software.

In practice, this use case may be the most challenging because of both the organizational challenges in enabling mutually distrusting entities to cooperate enough to develop the protocol execution system, and because of the complexity of the function to compute and the scale of the data. Nevertheless, specially optimized algorithms have been developed to perform secure private set intersection even in a malicious security model with reasonable performance over large data sets with millions to billions of records as discussed in Section 4.3.3.

5.1.4 Challenges for Secure Computation

We briefly mention here some additional unresolved issues as regards the use of any secure computation approach.

The first is the need to possibly edit errant inputs. This is done today with human interaction. Census Bureau analysts who have sworn status so they can access microdata examine the input data and attempt to correct any errors. Such a step is, inherently, not possible in a secure computation where the goal is to

prevent the input data from being visible to anyone but the owner. This raises a key issue of how to deal with the imperfections in data collection. If the data validation and correction processes can be expressed algorithmically, then they can be included in the MPC protocol and executed on the encrypted inputs. Another option might be to agree on a set of assertions that the data must satisfy, in terms of input as well as properties of intermediate results. These assertions could then be checked during the protocol execution. If any of these data quality and consistency assertions are not satisfied, the MPC would output an error. But it is important to ascertain that the error output does not disclose sensitive information about the inputs, so simply aborting during the computation is not usually an option. Finally, the output of the computation could be designed to include some additional data that could be used to perform sanity checks on the output and identify cases where there were problems with the data collection. Since this additional output would be revealed, it enables human inspection of the output and additional data, but it is also necessary to consider the additional output in the information disclosure analysis.

This issue is even more serious in cases where data providers may be motivated to deliberately provide incorrect input data to disrupt the computation, or cause the output produced to be incorrect in a particular way. For example, a business providing data may want to deceive its competitors, and can provide a false value to a Census Bureau survey that will result in misleading published statistics to achieve that goal. With the delegated MPC model (as suggested in Section 5.1.2), there is a possibility to allow the delegated data stewards to combine some data providers' inputs to detect dishonest providers. There could also be deployments in which the collected data is no longer sensitive after a certain time has passed, and then the data could be revealed to an analyst who would be able to retroactively detect errant providers.

A related issue is the requirement for MPC methods to facilitate imputation. When input data are missing, the Census Bureau uses various statistical models to impute the missing information. If this can be automated in some way, for example by applying machine learning or other statistical approaches, then, in principle, secure computation can be applied since any function of the data expressible as a finite Boolean circuit can be computed. However, whether common approaches to imputation such as hot-decking can be practically expressed in terms of a secure computation will require further investigation. There is work in this area applying decision trees that can be formulated as a secure computation [41].

Finally, further work is needed to understand how one archives data and results that originate from a secure computation. As discussed in Section 2.3, the Census Bureau is required to archive the "gold standard" data associated with its surveys. In so doing, data privacy must presumably be preserved. Archiving of secret-shared data is possible, but the protections MPC offers may be incompatible with regulatory archiving requirements. Archiving can either be done in a way that requires agreement from other operators to restore the data, or in a way that exposes the data thus giving up the privacy benefits of MPC. In any case, the implications of archiving requirements on secure computing should be explored.

5.2 Engendering Trust

Secure computing technologies can contribute to creating a trusted system for collecting, storing, and computing on data, but at best, are only part of the solution to guaranteeing the privacy of business data and to facilitating a more "friction-free" approach to performing various statistical investigations of the diverse datasets. The true basis of such efforts ultimately relies on the mutual trust placed in the various data stewards, and no cryptographic technology will change that. Secure computation can change how much trust in organizations and humans is required

for different aspects of a process, but that process still relies on trust in those organizations.

For example, the IRS must protect the privacy of Federal Tax Information and so must make sure that any statistical computations on such data do not leak information. Secure computing technologies can provide ways to perform those computations without leaking the inputs or intermediate results, but it is up to humans deploying those technologies to decide what functions can be computed, and what can be done with the outputs, as well as to ensure that the ways the technologies are used provide reasonable protections against likely threats.

In performing the Commodity Flow Survey or, for that matter, any survey relying on provision of data by various business establishments, participants need assurance that no business sensitive data will be released prematurely to the public or competitors. And in performing distributed record linkage of diverse data sets the relevant data stewards must all protect the privacy guarantees made to those who contributed the respective data. It is important that however secure computation technologies are used, the limitations of what they can provide are well understood, and no one views them as a panacea for addressing potential privacy risks. The Census Bureau has a long and distinguished record of protecting privacy in both its demographic and economic censuses, and so it is viewed as a trusted agency; this is the reason that the IRS and other agencies allow the Census Bureau to maintain copies of their data, and also in many cases to collect data on their behalf.

Confidence in the premise that an agency will provide the promised protections does not stem primarily from their technology choices. Certainly these are important, but there is more required. For example, when we deposit money in a bank by giving it to a teller or inserting it into an ATM, we are expressing an implicit trust in the bank. This is because the bank provides a host of signals that the establishment is physically secure, but also because there are a set of legal

guarantees that protect consumers should any corruption such as theft occur. Trust in data stewards has some similarities except corrective measures are much more limited—once data is leaked there is no way to reverse the loss in the way one has to recover funds lost due to a bank theft or default. The loss is one of public trust and this in turn has serious implications for the quality of future data products.

When a business or agency provides data to a data steward such as the Census Bureau, they are relying on many things, most of which are not technical. First, they must rely on the security of their own computer and software infrastructure, in particular its interactions over a network. This requires technology, but it also requires the responsiveness of their IT departments. On the data steward side there is a similar reliance on computers and software, but also physical security and IT policies, and on the honesty and competence of the people responsible for administering them. It is also necessary to trust the various committees who establish data policies and finally, that there is a legal framework that allows some recourse should all the above protections prove insufficient.

Secure computing technologies provide value when they can reduce complexity by focusing questions on what must be trusted. For example, trusted execution environments modify the trust environment from the requirement to fully trust a large and complex software and hardware stack, to trusting that a secure enclave is implemented in a way that provides the claimed security properties, that the vendor responsible for the attestation keys is honest and manages keys correctly, and that the program running in the enclave, which is hopefully small enough to audit, is correct and does not produce outputs that would leak sensitive information about the inputs. Ensuring such properties is challenging, but the task becomes better defined than what is required in the absence of a trusted execution environment. When decisions about what programs can be run in the enclave (or, similarly, in a multi-party computation) are delegated to others, then the data providers need to trust the organizations and processes responsible for making

those decisions just as much as they would absent the use of secure computing technologies. At a trivial level, a query that succeeds in exposing an individual's tax information is insecure regardless of the sophistication of the secure computing technology being employed.

5.3 Responses to the Sponsor's Questions

Is the Census Bureau researching the technologies best suited to the purpose?

The Census Bureau is researching the technologies best suited to the purpose. Secure computation technologies could play an important role in improving security and enhancing mutual trust among statistical agencies and data providers.

What investments might make MPC technology operate at scale for a suite of business statistics?

There are available today several open source and commercial offerings for MPC technologies. Current MPC offerings from commercial providers are not immediately suited to Census Bureau applications, and some analysis and development would be required to support the scale needed for some Census Bureau applications like distributed record linkage for large data. Previous MPC deployments, such as those accomplished by the Estonian government and by Google and its advertisers, as well as academic results on applications like private set intersection and genomic analysis, do demonstrate the potential for using these technologies at the scale needed by the Census Bureau.

The investment that would most likely make secure computation technology beneficial to the Census Bureau and operate at a scale relevant to Census Bureau application would be the development of a collaboration among the Census Bureau and a set of experts in MPC, both in industry and academia with the objective of setting up a set of pilot projects. We discuss this further in our recommenda-

tions below. Ultimately, if these pilot projects are successful, the next step would be building an in-house team of developers to evaluate MPC designs and develop special purpose MPC protocols for particular operations needed to achieve the scalability required for the Census Bureau.

If MPC is chosen to perform the computation in a Delegated MPC model where the Census Bureau operates one of the servers and another agency operates the other server, data providers may be more willing to participate in data collection because of the reduced reliance on a single data collector and because the trust is now distributed over multiple data collectors working securely in concert. Data providers still need to trust the security and correctness of their own hardware and software, but now only need to trust that the delegated server operators do not collude, and that the processes they use to decide what computations can be run on the data are administered responsibly. The data providers would be trusting that the implementations of cryptography and the various protocols are correct and provide the expected security properties.

Are there medium-range feasible SMC tools that could be used to enhance applications using record-level linkage without ingesting the full supplemental database?

Many record-level linkage problems can be viewed as private set intersection with some computation on the intersection. There are known algorithms to perform private set intersection very efficiently using custom MPC protocols that can already scale to sets with billions of elements (see Section 4.3.3). These approaches cannot be used directly for all the types of record linkage required for applications like OnTheMap since they can only perform exact equality tests in the join and the computations that can be performed on the intersection are limited (e.g., sum), but they may provide a framework upon which such applications can be built.

Would chaining SGX enclaves in a commercial cloud environment support complex processing on a scale that meets the Census Bureau's needs?

Trusted execution environments are widely deployed using secure hardware enclaves implemented by Intel, AMD and others; the major cloud providers employ such hardware to provide secure computing services in data centers. In principle, given the current computational throughput of Intel's SGX enclaves, it is plausible that chaining SGX enclaves in a commercial cloud environment could support processing on a scale that meets the Census Bureau's needs. However, as we argue in Section 4.4.1, the current vulnerabilities and fundamental security challenges associated with such enclaves mean that the claimed security properties they tout are unlikely to be achieved in practice. Since SGX cannot provide security in its intended threat model, it is necessary to establish the security of the host system and data center also, and if the security of these components can be established the additional benefits of SGX are in question. As discussed in Section 4.4.2, JASON has serious concerns about the security of SGX in the threat model for which it is designed. Because of the present-day security vulnerabilities of SGX, it would be necessary to verify the physical and software security of all the cloud environments being utilized. The secure enclave approach as currently implemented does not provide much more security than what is obtained using conventional security approaches for cloud infrastructure.

Does Secure Multiparty Computation (SMC) offer an opportunity to reduce burden on companies while continuing to provide the needed economic data?

There are promising opportunities to use SMC and, in particular, MPC to collect and process data in a way that would address security and privacy concerns of companies providing data and that may allow them to participate in surveys like the Commodity Flow Survey in a more direct way through automated interaction with those companies' data repositories. The best way to answer this question

would be to undertake a pilot study as discussed in the recommendations below to determine if the use of MPC will enhance mutual trust among data providers and the Census Bureau and also provide a means of communicating data to the Census Bureau that reduces respondent burden.

How do we build trust in SMC so that companies are willing to participate?

As discussed above, engendering trust in a secure computing system will require verification that the proposed use of SMC does indeed provide the requisite security, and that measures of verification of that technology are also trustworthy. Such measures include code reviews of the various queries and demonstrations that collusion among any parties can be deterred or detected. Finally, an issue for further research is whether it is possible to use formal methods to verify the correctness of an MPC implementation, and to use automated tools to examine the queries to be dispatched via SMC for potential disclosure. Such an approach might be useful in automating whether a proposed query might leak sensitive information, and may reduce some of the dependence on human review. This is a challenging problem, but if some progress along such lines could be made it would both streamline the process of processing the data collected, and increase confidence that a program executed using SMC will release only that information that is mutually agreed upon.

Are there other conceptual approaches we should investigate?

JASON believes the proposed investigation of secure computation technologies is the right first step.

5.4 Findings

The following are general findings:

1. Secure computation technologies are not a substitute for query and disclosure avoidance analysis. It is essential that any functions to be computed be reviewed for potential disclosure of sensitive information, and that all implementations be carefully reviewed for potential leaks. Additionally, secure computation cannot in and of itself be the determining factor in deciding if a query or calculation satisfies a required statutory benefit.

2. Secure computation technologies provide some opportunities for replacing trust in humans and processes with trust in technical solutions, but can only partially reduce the need to trust humans and processes. In any potential deployment, it is important to consider the required security and disclosure properties, and to understand who or what is responsible for different aspects of ensuring them.

Our finding on fully homomorphic encryption is as follows:

3. Fully Homomorphic Encryption (FHE) is unlikely to be of use for any Census Bureau application in the foreseeable future. The computational costs associated with FHE are prohibitive for any Census-scale computation, and, although there has been rapid progress in reducing computational costs over the past several years, major breakthroughs would be needed before FHE becomes practical for the types of applications currently under consideration by the Census Bureau.

The following are our findings on multi-party computation (MPC):

4. MPC is a well-studied although not fully mature technology with tools

available for building MPC applications, successful deployments, and well-understood security properties.

5. MPC can be used to compute any function expressible as a circuit securely, but the costs, dominated by network bandwidth, of general purpose MPC are high.

6. For some computations, sufficiently efficient MPC solutions are known that can scale to billions of inputs, including solutions for tabulation and private set intersection with aggregation.

7. Using MPC securely requires attention to implementation details. It can be used to eliminate the need to trust other participants in the computation, but does not absolve one of the need to review the disclosure risks of the function to be computed, or the need to trust one's own software and hardware.

8. By design, using MPC means that data inputs are not visible for human review during the computation. Any data editing procedures must either be fully automated or performed by the data providers or perhaps a trusted third party.

9. Imputation of missing data can be performed within MPC in principle, but further work will be required to determine whether it can be performed for applications at Census scale.

The following are our findings on use of secure enclaves, in particular, Intel SGX:

10. SGX is designed for a very specific threat model: running code on a host where the operating system could be compromised. In principle, it enables running code in the secure enclave without exposing any data under protection of the enclave to the compromised host.

11. SGX currently does not provide the promised protection, and has been vulnerable to known exploits for its entire history. Recent vulnerabilities not only allow leakage of data from a single enclave, but enable forged attestations of the enclave state, completely breaking the security premises of the SGX ecosystem.

12. If a sound and secure enclave could be implemented that satisfies the goals of SGX, it could allow for a self-contained program to be audited for desired disclosure properties, and for external users to verify and validate that this is the only program that will have access to data deemed sensitive.

13. Adapting a program to run securely within SGX (or other Trusted Execution Environments) requires specialized expertise, the absence of which increases the risk that sensitive information is leaked.

14. At present, SGX provides no clear advantage over the trusted data steward model for the use cases of interest to the Census Bureau.

15. New, potentially more secure designs for hardware enclaves are under development, but are not likely to become widely available within a five year time horizon.

The following are our findings on the use of secure computation technologies for applications of interest to the Census Bureau:

16. There are potential uses of MPC technologies for two types of Census Bureau applications:

    - The Delegated MPC model may provide an improved expectation of trust when the Census Bureau wants to collect and analyze data from providers who may be reluctant to participate owing to concerns of security and confidentiality.

- For applications where the Census Bureau and other agencies want to compute jointly on separately-stewarded data, the Joint Data Provider MPC model may provide a way to satisfy regulatory constraints with less reliance on inter-agency trust.

17. Three aspects of Census Bureau applications could pose challenges for the use of MPC technologies:

    - First, using MPC means that the data inputs are not visible for human review. Any data editing procedures must either be fully automated or performed by the data providers themselves. Any auditing of provided data or mechanisms to detect misbehaving data providers must also be fully automated, so algorithms used to aggregate such data much be resilient to inputs that may be provided maliciously.
    - Second, imputation of missing data can be performed within MPC in principle, but, depending on the specifics of the imputation method and the size of the data, it may be practically infeasible. Further research is required regarding this issue.
    - Third, archiving of secret-shared data is possible, but the protections afforded by MPC may be incompatible with regulatory archiving requirements. Archiving must either be performed in a way that preserves data privacy, thus requiring agreement among the data providers when the data are restored, or in a way that combines the data, thus giving up the privacy properties provided by MPC.

5.5 Recommendations

JASON's recommendations are as follows:

1. JASON recommends that the Census Bureau undertake with BEA a pilot study to consider the use of the Single Steward model of MPC in order

to link confidential microdata. The initial pilot should be done using data fully accessible to the Census Bureau to gain experience without any cross-organization complexities. The initial computation should be specifiable as a simple program (i.e., code fits on one page) that can be effectively reviewed for potential query disclosure. If successful, a subsequent pilot should be undertaken in partnership with an external data provider and a simple joint computation should be performed.

2. JASON recommends that the Census Bureau undertake a pilot study to consider the use of the Delegated MPC model in performing the Commodity Flow Survey. A key goal of this study should be to investigate whether such an approach could mitigate privacy concerns for businesses as regards sharing of confidential business data with the Census Bureau, enhance trust (in particular whether there are mutually trusted organizations that would partner with the Census Bureau in the MPC), and whether collecting data with privacy guarantees would reduce respondent burden by allowing the respondents to be less selective about the data they provide.

3. JASON recommends that Census Bureau investigate tests for potential disclosure of sensitive information by a query so that certain types of queries can be formally specified with sufficient precision to implement as algorithms without the need for human review. Such automation will streamline the processing of queries for all MPC models as well the Single Steward model. Automating query approval would have substantial benefits if possible, but poses both regulatory and technical challenges that would require further study.

This Page Intentionally Left Blank

References

[1] Emmanuel A Abbe, Amir E Khandani, and Andrew W Lo. Privacy-preserving methods for sharing financial risk exposures. *American Economic Review*, 102(3):65–70, 2012.

[2] Frederik Armknecht, Colin Boyd, Christopher Carr, Kristian Gjøsteen, Angela Jäschke, Christian A Reuter, and Martin Strand. A guide to fully homomorphic encryption. *IACR Cryptology ePrint Archive*, 2015:1192, 2015.

[3] Gilad Asharov and Claudio Orlandi. Calling out cheaters: Covert security with public verifiability. In *AsiaCrypt*, 2012.

[4] Yonatan Aumann and Yehuda Lindell. Security against covert adversaries: Efficient protocols for realistic adversaries. In *Theory of Cryptography Conference*, 2007.

[5] Donald Beaver. Efficient multiparty protocols using circuit randomization. In *CRYPTO*, pages 420–432, 1992.

[6] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *ACM Symposium on Theory of Computing*, pages 1–10, 1988.

[7] J. Benedetto, G. Stanley and E. Totty. The creation and use of the SIPP synthetic beta v7.0. *CES Technical Notes Series 18-03*, 2018.

[8] Dan Bogdanov. *Sharemind: programmable secure computations with practical applications*. PhD thesis, University of Tartu, 2013.

[9] Dan Bogdanov, Liina Kamm, Baldur Kubo, Reimo Rebane, Ville Sokk, and Riivo Talviste. Students and taxes: a privacy-preserving study using secure computation. *Proceedings on Privacy Enhancing Technologies*, 2016(3):117–135, 2016.

[10] Thomas Bourgeat, Ilia A. Lebedev, Andrew Wright, Sizhuo Zhang, Arvind, and Srinivas Devadas. MI6: secure enclaves in a speculative out-of-order processor. *CoRR*, abs/1812.09822, 2018.

[11] David Chaum. Blind signature system. In *CRYPTO*, 1983.

[12] Raj Chetty et al. The opportunity atlas. `https://www.opportunityatlas.org`, accessed August 14, 2020.

[13] Raj Chetty et al. Economic tracker. `https://opportunityinsights.org/tracker-resources`, accessed September 29, 2020.

[14] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. TFHE: fast fully homomorphic encryption over the torus. *J. Cryptology*, 33(1):34–91, 2020.

[15] Hyunghoon Cho, David J Wu, and Bonnie Berger. Secure genome-wide association analysis using multiparty computation. *Nature Biotechnology*, 36(6):547–551, 2018.

[16] Jonan Chu and Daniel DeGraaf. Confidently computing with confidential computing. Presentation to JASON June 17, 2020.

[17] Tobias Cloosters, Michael Rodler, and Lucas Davi. TeeRex: Discovery and exploitation of memory corruption vulnerabilities in SGX enclaves. In *29th USENIX Security Symposium*, 2020.

[18] V. Costan and S. Devadas. Intel SGX explained. *Cryptology ePrint Archive*, 2016.

[19] Victor Costan, Ilia Lebedev, and Srinivas Devadas. Sanctum: Minimal hardware extensions for strong software isolation. In *25th USENIX Security Symposium*, 2016.

[20] DARPA. DPRIVE. `https://www.darpa.mil/attachments/DPRIVEProposersDaybriefing20200227.pdf`, accessed August 14, 2020.

[21] Irit Dinur and Kobbi Nissim. Revealing information while preserving privacy. In *Symposium on Principles of Database Systems*, pages 202–210. ACM, 2003.

[22] Jack Doerner, David Evans, and Abhi Shelat. Secure stable matching at scale. In *ACM Conference on Computer and Communications Security*, 2016.

[23] Jack Doerner and Abhi Shelat. Scaling ORAM for secure computation. In *ACM Conference on Computer and Communications Security*, pages 523–535, 2017.

[24] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Third Conference on Theory of Cryptography*, 2006.

[25] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*, 31(4):469–472, 1985.

[26] David Evans, Vladimir Kolesnikov, and Mike Rosulek. *A Pragmatic Introduction to Secure Multi-Party Computation*. NOW Publishers, 2018.

[27] Jun Furukawa and Yehuda Lindell. Two-thirds honest-majority MPC for malicious adversaries at almost the cost of semi-honest. In *ACM Conference on Computer and Communications Security*, 2019.

[28] Adrià Gascón, Phillipp Schoppmann, Borja Balle, Mariana Raykova, Jack Doerner, Samee Zahur, and David Evans. Privacy-preserving distributed linear regression on high-dimensional data. *Proceedings on Privacy Enhancing Technologies*, 2017.

[29] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the Forty-first annual ACM Symposium on Theory of Computing*, pages 169–178, 2009.

[30] Oded Goldreich. *Foundations of Cryptography: Volume 2*. Cambridge University Press, 2004.

[31] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In *ACM Symposium on Theory of Computing*, pages 218–229, 1987.

[32] Oded Goldreich and Rafail Ostrovsky. Software protection and simulation on Oblivious RAMs. *Journal of the ACM*, 43(3), 1996.

[33] S. Dov Gordon, Jonathan Katz, Vladimir Kolesnikov, Fernando Krell, Tal Malkin, Mariana Raykova, and Yevgeniy Vahlis. Secure two-party computation in sublinear (amortized) time. In *ACM Conference on Computer and Communications Security*, pages 513–524, 2012.

[34] Matthew Graham. LODES/OnTheMap. Presentation to JASON June 18, 2020.

[35] Trinabh Gupta, Henrique Fingler, Lorenzo Alvisi, and Michael Walfish. Pretzel: Email encryption and provider-supplied functions are compatible. In *Conference of the ACM Special Interest Group on Data Communication (SIGCOMM)*, 2017.

[36] Marcella Hastings, Brett Hemenway, Daniel Noble, and Steve Zdancewic. SoK: General purpose compilers for secure multi-party computation. In *IEEE Symposium on Security and Privacy*, 2019.

[37] Brett Hemenway, Steve Lu, Rafail Ostrovsky, and William Welser. High-precision secure computation of satellite collision probabilities. In *International Conference on Security and Cryptography for Networks*, 2016.

[38] Yan Huang, David Evans, and Jonathan Katz. Private set intersection: Are garbled circuits better than custom protocols? In *Network and Distributed Systems Security Symposium*, 2012.

[39] Mihaela Ion, Ben Kreuter, Ahmet Erhan Nergiz, Sarvar Patel, Mariana Raykova, Shobhit Saxena, Karn Seth, David Shanahan, and Moti Yung. On deploying secure computing: Private

intersection-sum-with-cardinality. Cryptology ePrint Archive, Report 2019/723, 2019. https://eprint.iacr.org/2019/723.

[40] Yuval Ishai, Joe Kilian, Kobbi Nissim, and Erez Petrank. Extending oblivious transfers efficiently. In *CRYPTO*, pages 145–161, 2003.

[41] Geetha Jagannathan and Rebecca N. Wright. Privacy-preserving imputation of missing data. *Data Knowl. Eng.*, 65(1):40–56, April 2008.

[42] Chiraag Juvekar, Vinod Vaikuntanathan, and Anantha Chandrakasan. Gazelle: A low latency framework for secure neural network inference. In *27th USENIX Security Symposium*, 2018.

[43] Paul Kocher, Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, Michael Schwarz, and Yuval Yarom. Spectre attacks: Exploiting speculative execution. *CoRR*, abs/1801.01203, 2018.

[44] Vladimir Kolesnikov and Alex J. Malozemoff. Public verifiability in the covert model (almost) for free. In *AsiaCrypt*, pages 210–235, 2015.

[45] Vladimir Kolesnikov and Thomas Schneider. Improved garbled circuit: Free XOR gates and applications. In *International Colloquium on Automata, Languages and Programming*, pages 486–498, 2008.

[46] Benjamin Kreuter. Secure Multiparty Computation at Google. Real World Crypto (https://www.youtube.com/watch?v=ee7oRsDnNNc), 2017.

[47] A. Lapets, N. Volgushev, A. Bestavros, F. Jansen, and M. Varia. Secure MPC for analytics as a web application. In *IEEE Cybersecurity Development (SecDev)*, 2016.

[48] Andrei Lapets, E. Dunton, Kyle Holzinger, Frederick Jansen, and Azer Bestavros. Web-based multi-party computation with application to anonymous aggregate compensation analytics. http://www.cs.bu.edu/techreports/pdf/2015-009-mpc-compensation.pdf, 2015.

[49] Andrei Lapets, Nikolaj Volgushev, Azer Bestavros, Frederick Jansen, and Mayank Varia. Secure multi-party computation for analytics deployed as a lightweight web application. Technical report, Boston University, 2016.

[50] Berin Linfors, Grant Degler, and Christian Moscardi. Commodity flow survey. Presentation to JASON June 17,, 2020.

[51] Moritz Lipp, Michael Schwarz, Daniel Gruss, Thomas Prescher, Werner Haas, Anders Fogh, Jann Horn, Stefan Mangard, Paul Kocher, Daniel Genkin, Yuval Yarom, and Mike Hamburg. Meltdown: Reading kernel memory from user space. In *27th USENIX Security Symposium*, 2018.

[52] Jian Liu, Mika Juuti, Yao Lu, and Nadarajah Asokan. Oblivious Neural Network predictions via MiniONN transformations. In *ACM Conference on Computer and Communications Security*, 2017.

[53] Ashwin Machanavajjhala, Daniel Kifer, John Abowd, Johannes Gehrke, and Lars Vilhuber. Privacy: Theory meets practice on the map. In *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering*, ICDE '08, page 277–286, USA, 2008. IEEE Computer Society.

[54] Daniele Micciancio. Fully homomorphic encryption from the gorund up. `http://cseweb.ucsd.edu/~daniele/papers/FHEeurocrypt19.pdf`, 2019.

[55] Payman Mohassel and Yupeng Zhang. SecureML: A system for scalable privacy-preserving machine learning. In *IEEE Symposium on Security and Privacy*, 2017.

[56] Kit Murdock, David Oswald, Flavio D. Garcia, Jo Van Bulck, Daniel Gruss, and Frank Piessens. Plundervolt: Software-based fault injection attacks against Intel SGX. In *41st IEEE Symposium on Security and Privacy (S&P'20)*, 2020.

[57] Michael Naehrig, Kristin Lauter, and Vinod Vaikuntanathan. Can homomorphic encryption be practical? In *Proceedings of the Third ACM workshop on Cloud computing security*, pages 113–124, 2011.

[58] Valeria Nikolaenko, Udi Weinsberg, Stratis Ioannidis, Marc Joye, Dan Boneh, and Nina Taft. Privacy-preserving ridge regression on hundreds of millions of records. In *IEEE Symposium on Security and Privacy*, 2013.

[59] Dept. of Transportation. Commodity flow survey. `https://www.bts.gov/cfs`, accessed September 29, 2020.

[60] Rafail Ostrovsky and Victor Shoup. Private information storage (extended abstract). In *ACM Symposium on Theory of Computing*, pages 294–303, 1997.

[61] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *IN ADVANCES IN CRYPTOLOGY — EUROCRYPT 1999*, pages 223–238. Springer-Verlag, 1999.

[62] Benny Pinkas, Mike Rosulek, Ni Trieu, and Avishay Yanai. PSI from PaXoS: Fast, malicious private set intersection. In Anne Canteaut and Yuval Ishai, editors, *EuroCrypt*, 2020.

[63] Oded Regev. The learning with errors problem (invited survey). In *Proceedings of the 2010 IEEE 25th Annual Conference on Computational Complexity*, CCC '10, page 191–204, USA, 2010. IEEE Computer Society.

[64] M Sadegh Riazi, Mohammad Samragh, Hao Chen, Kim Laine, Kristin Lauter, and Farinaz Koushanfar. XONN: XNOR-based oblivious deep neural network inference. In *28th USENIX Security Symposium*, 2019.

[65] Ronald L Rivest, Len Adleman, and Michael L Dertouzos. On data banks and privacy homomorphisms. *Foundations of Secure Computation*, 4(11):169–180, 1978.

[66] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[67] Sajin Sasy, Sergey Gorbunov, and Christopher W Fletcher. ZeroTrace: Oblivious memory primitives from intel SGX. In *Network and Distribute Systems Security Symposium*, volume 2018, 2017.

[68] Internal Revenue Service. IRS form 1040. https://www.irs.gov/pub/irs-pdf/f1040.pdf, accessed September 29, 2020.

[69] Internal Revenue Service. IRS form 1065. https://www.irs.gov/pub/irs-access/f1065_accessible.pdf, accessed September 29, 2020.

[70] Internal Revenue Service. IRS form 1120. https://www.irs.gov/pub/irs-pdf/f1120.pdf, accessed September 29, 2020.

[71] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[72] Abhi Shelat and Chih-Hao Shen. Two-output secure computation with malicious adversaries. In *EuroCrypt*, 2011.

[73] Erich Strassner and Tina Highfill. Linking confidential federal microdata to calculate economic statistics by enterprise size: Jason briefing. Presentation to JASON June 17,, 2020.

[74] Unbound Tech, Inc. NextGen vHSM (Virtual Hardware Security Module). https://www.unboundtech.com/virtual-hardware-security-module/.

[75] United States Census Bureau. Business register. https://www.census.gov/econ/overview/mu0600.html, accessed August 14, 2020.

[76] United States Census Bureau. OnTheMap. https://onthemap.ces.census.gov, accessed August 14, 2020.

[77] United States Census Bureau. Statistics of US businesses. https://www.census.gov/programs-surveys/susb.html, accessed August 14, 2020.

[78] United States Census Bureau. *Survey of Income and Program Participation*. U.S. Census Bureau, accessed August 15, 2020.

[79] United States Census Bureau. *Synthetic SIPP Data*. U.S. Census Bureau, accessed August 15, 2020.

[80] United States Code. *Title 26, §6103*. U.S.C., 1986.

[81] United States Code. *Title 13, §§8, 9, 23(c), 401, 402*. U.S.C., 1990.

[82] United States Code. *Title 22, §3104(c)*. U.S.C., 1990.

[83] United States Code. *Title 44, §3561, et. seq.* U.S.C., 2017.

[84] U.S. Bureau of Labor Statistics. Quarterly census of employment and wages. `https://www.bls.gov/cew/`, accessed September 29, 2020.

[85] U.S. Bureau of Transportation Statistics. Standard classification of transported goods. `https://www.bts.gov/archive/publications/commodity_flow_survey/classification`, accessed September 29, 2020.

[86] U.S. Census Bureau. American community survey. `https://www.census.gov/programs-surveys/acs`, accessed September 29, 2020.

[87] U.S. Census Bureau. Decennial census of population and housing. `https://www.census.gov/programs-surveys/decennial-census.html`, accessed September 29, 2020.

[88] U.S. Census Bureau. Longitudinal-employer household dynamics. `https://lehd.ces.census.gov/`, accessed September 29, 2020.

[89] U.S. Census Bureau. North american industry classification system. `https://www.census.gov/eos/www/naics/`, accessed September 29, 2020.

[90] U.S. Census Bureau and Internal Revenue Service. Agreement for the Review and Approval of U. S. Census Bureau Projects that use Federal Tax Information, U.S. Census Bureau-International Revenue Service Research, June 2012.

[91] U.S. Census Bureau and National Archives and Records Administration (NARA). *Memorandum of Understanding*. U.S. Census and NARA, Febuary 2008.

[92] U.S. Census Bureau, Data Stewardship Executive Policy Committee. *Policy on Title 13 Benefit Statements DS002*. Policy Coordination Office, 2018.

[93] Jo Van Bulck, Marina Minkin, Ofir Weisse, Daniel Genkin, Baris Kasikci, Frank Piessens, Mark Silberstein, Thomas F. Wenisch, Yuval Yarom, and Raoul Strackx. Foreshadow: Extracting the keys to the Intel SGX kingdom with transient out-of-order execution. In *27th USENIX Security Symposium*, SEC'18, page 991–1008, USA, 2018. USENIX Association.

[94] Stephan van Schaik, Andrew Kwong, Daniel Genkin, and Yuval Yarom. SGAxe: How SGX fails in practice. `https://sgaxeattack.com/`, 2020.

[95] Stephan van Schaik, Marina Minkin, Andrew Kwong, Daniel Genkin, and Yuval Yarom. Cacheout: Leaking data on Intel cpus via cache evictions, 2020.

[96] Xiao Shaun Wang, Yan Huang, Yongan Zhao, Haixu Tang, XiaoFeng Wang, and Diyue Bu. Efficient genome-wide, privacy-preserving similar patient query based on private edit distance. In *ACM Conference on Computer and Communications Security*, pages 492–503, 2015.

[97] David J Wu, Tony Feng, Michael Naehrig, and Kristin Lauter. Privately evaluating decision trees and random forests. *Proceedings on Privacy Enhancing Technologies*, 2016(4):335–355, 2016.

[98] Samee Zahur, Mike Rosulek, and David Evans. Two Halves Make a Whole: Reducing Data Transfer in Garbled Circuits using Half Gates. In *Advances in Cryptology–EUROCRYPT 2015*. Springer, 2015.

[99] Yanjun Zhang, Guangdong Bai, Xue Li, Caitlin Curtis, Chen Chen, and Ryan K L Ko. PrivColl: Practical privacy-preserving collaborative machine learning. In *25th European Symposium on Research in Computer Security*, 2020.

[100] Ruiyu Zhu, Yan Huang, Jonathan Katz, and Abhi Shelat. The cut-and-choose game and its application to cryptographic protocols. In *USENIX Security Symposium*, pages 1085–1100, 2016.