Secure MPC as a Tool for Sensitive Data Analysis

Samuel Havron ⟨havron@virginia.edu⟩

March 21, 2016

1 Introduction

A *Secure multi-party computation* (MPC) is a protocol which allows for two or more parties to compute a function on sensitive input data provided by each party, without revealing anything about the inputs (other than what can be inferred from the revealed output result).

Most current implementations of MPC work by executing instructions in a *data-oblivious* manner, where the control flow of the program is independent of the inputs provided by each party and the program executes without any knowledge of the cleartext data it is operating on.

- Capabilities of MPC (including overview of previous work, Samee's implementation, efficiency/speed comparison with non-secure MPC programs)
- Value of MPC to social scientists, researchers (with motivating example(s)) as a tool for sensitive data analysis

For instance, an education researcher may want to perform linear regression analysis on middle school students' GPAs and their family's income, without revealing any sensitive data about the students or their families. In one example, an education researcher may be interested in using statistical methods to analyze the relationship between a student's family income and the student's Grade Point Average for a particular school system or collection of school systems. Obtaining such sensitive data ... (* reference to CMU 2009 paper *)

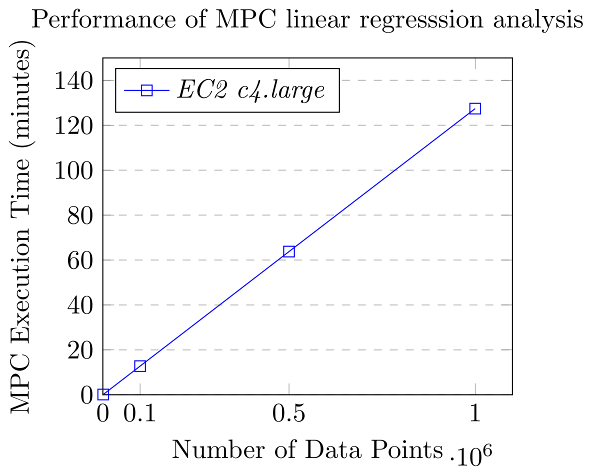
2 Methods

- Description of OblivCs efficiency/speed in comparison to other existing MPC implementations
- Overview of OblivC as a language, relationship to C (appeal to researchers with some C/basic programming experience)
- Introduce implementation of linear regression analysis program, use code snippets and reference repo/site tutorial

3 Results

- Discuss efficiency/speed/scalability of aforementioned linear regression analysis program, provide EC2 testing data (collect more formally prior to including in final paper). Mention single-threaded implementation and limits on performance results.
- Discuss results of other motivating OblivC programs, compare to alternative MPC results

The scalability and speed of OblivC as a tool for implementing MPC programs was tested using c4.large *Elastic Compute Cloud* (EC2) nodes from *Amazon Web Services* (AWS) of Amazon.com®. Two c4.large instances were launched and connected through OblivC's API for TCP/IP connections. One node instance provided independent (x) data points, while the other provided dependent (y) data points. Data points used were 32-bit integers. Data for computation was mined from New York public health data (include link), as well as synthetic data points generated through a Python program (footnote for file location) for performance results.



- Include table with times from NY public health dataset computation.
- Include table with times from non secure MPC computation for comparison.

4 Conclusion

- Overview of secure MPC, utility/extensibility of OblivC programming, real-world usage for scientists/researchers

A secure MPC ensures that only the result of data analysis on sensitive inputs is revealed. OblivC implements this under-the-hood using manipulations of Yao's Garbled Circuits Protocol (* link to Samee's paper *).

Using OblivC as a tool to analyze real, sensitive data-sets requires only a basic understanding of the C programming language, and enthusiasm for experimenting with new API that OblivC provides. A full tutorial for how to get started and an API documentation are available online (* update with link *). One particular aspect of OblivC in its current release is the lack of native support for oblivious-qualified floating point numbers. A workaround using fixed point numbers is detailed in the linear regression example, specifically here (* update with link *). Preliminary work has been included in the code repository (* link *) for automating a data-matching process between two datasets, where a common identifier for data values (e.g. SSN) is mapped and party inputs are matched up with a private set intersection protocol; researchers may find modifications to this source code useful in creating their own secure MPC applications for matching sensitive datasets with a common identifier.

References

[1] Yehuda Lindell and Benny Pinkas, *Secure Multiparty Computation for Privacy-Preserving Data Mining*, The Journal of Privacy and Confidentiality (2009), Number 1, pp. 5998. Retrieved from http://repository.cmu.edu/cgi/viewcontent.cgi?article=1004&context=jpc.