Tutorial: Deploying Secure Multi-Party Computation on the Web Using JIFF

Kinan Dak Albab, Rawane Issa, Andrei Lapets, Peter Flockhart, Lucy Qin, Ira Globus-Harris
Boston University, 111 Cummington Mall, Boston, MA USA 02215
Email: {babman, ra1issa, lapets, pflock, lucyq, igh} @bu.edu

I. Introduction

Secure multi-party computation (MPC) is a cryptographic primitive that enables several parties to jointly compute over their collective private data sets. Over the past decade, a number of general and special-purpose MPC software frameworks have been developed [1]. Many of them have certain characteristics that can hinder their adoption (and consequently the adoption of MPC itself) in practical real-world applications:

1) Due to their domain-specific languages, frameworks are difficult to use within larger applications [2], [3].
2) They often do not provide easy-to-use abstractions for expressing asymmetries between participating parties.
3) They are often tied to the underlying assumptions that are used to optimize their protocols.
4) Deploying and using MPC applications built with these frameworks requires substantial technical expertise.

The JavaScript Implementation of Federated Functionalities (JIFF) is an MPC framework built specifically to address the above concerns. JIFF is built using JavaScript to support MPC applications that (1) run on web and mobile platforms in which parties join and leave the computation dynamically, (2) require that computations occur asynchronously, and (3) require mechanisms for recovery from network and crash failures. JIFF is highly customizable to accommodate the idiosyncrasies of deployment scenarios, and can be easily integrated within server-client(s) web and mobile applications, and server(s)-to-server(s) systems, in which the essential non-MPC features have been built by developers who do not necessarily possess extensive cryptographic know-how.

JIFF has been used to implement several MPC components and applications, including libraries for data structures, machine learning, and relational data workflows. JIFF has been deployed in two ongoing real-world privacy-preserving studies [4], [5]: assessing the wage gap in partnership with the Boston Women's Workforce Council, and evaluating corporate spending toward women and minority-owned businesses in partnership with the Greater Boston Chamber of Commerce.

II. Tutorial

The tutorial introduces the basics of MPC, including practical background knowledge on its core security guarantees, and some of its deployed use cases in practice. To motivate some of the design decisions made while building JIFF, we ask participants to partake in a pre-deployed MPC demo available on the web through browsers and mobile phones.

The bulk of the tutorial overviews the JIFF library and its use for building server-assisted web MPC applications, by going through selected code snippets and demos interactively. We cover topics ranging from the basic setup and programming API, to customizing JIFF's protocols and hooking it to larger web applications. We contrast two main paradigm differences between MPC programs and their insecure counterpart: (1) the encoding of control flow and branching statements and (2) the performance trade-offs between traditionally-similar operators. We showcase general tips for improving performance through parallelization, lazy-evaluation, and pre-processing. All snippets and demos are available in the JIFF repository `https://github.com/multiparty/jiff`.

III. Audience and Outcomes

This tutorial is open to anyone interested in learning how to build and deploy secure multi-party computation within the context of a web application. All levels of experience are welcome. Since the tutorial entails writing code in JavaScript and Node.js, audience members should at a minimum have familiarity with JavaScript.

Acknowledgments

We acknowledge the various JIFF developers and contributors listed at `multiparty.org`. JIFF itself is supported by NSF grants #1414119, #1718135, and #1739000.

References

[1] M. Hastings, B. Hemenway, D. Noble, and S. Zdancewic, "Sok: General purpose compilers for secure multi-party computation," in *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019.

[2] D. Bogdanov, S. Laur, and J. Willemson, "Sharemind: A framework for fast privacy-preserving computations," in *Computer Security - ESORICS 2008*, S. Jajodia and J. Lopez, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 192–206.

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

[4] A. Lapets, N. Volgushev, A. Bestavros, F. Jansen, and M. Varia, "Secure mpc for analytics as a web application," in *2016 IEEE Cybersecurity Development (SecDev)*, Nov 2016, pp. 73–74.

[5] A. Lapets, F. Jansen, K. D. Albab, R. Issa, L. Qin, M. Varia, and A. Bestavros, "Accessible privacy-preserving web-based data analysis for assessing and addressing economic inequalities," in *Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies*, ser. COMPASS '18. Menlo Park and San Jose, CA, USA: ACM, 2018, pp. 48:1–48:5. [Online]. Available: http://doi.acm.org/10.1145/3209811.3212701

978-1-5386-7289-1/19/$31.00 ©2019 IEEE
DOI 10.1109/SecDev.2019.00013

IEEE computer society
Authorized licensed use limited to: BOSTON UNIVERSITY. Downloaded on November 14,2025 at 23:26:19 UTC from IEEE Xplore. Restrictions apply.