Secure Multiparty Computation for Cooperative Cyber Risk Assessment

Kyle Hogan, Noah Luther, Nabil Schear, Emily Shen, David Stott, Sophia Yakoubov, Arkady Yerukhimovich
MIT Lincoln Laboratory
Emails: {kyle.hogan, noah.luther, nabil, emily.shen, david.stott, sophia.yakoubov, arkady}@ll.mit.edu

A common problem organizations face is determining how to prioritize security updates and patches to minimize the risk of vulnerabilities in their infrastructure. Limited resources constrain organizations to select a set of only the most security critical updates that they can afford to perform; thus, it is very important to compute the risk of delaying less critical updates accurately [9]. Organizations can improve the accuracy of their cyber risk assessments by pooling their data in order to account for attacks that others have experienced [7]. However, privacy concerns may prevent this type of information sharing, as organizations are understandably unwilling to reveal details pertaining to current vulnerabilities or past attacks, which could be damaging to both their security and their reputation.

We propose the use of secure multiparty computation (MPC) to allow organizations to perform joint analytics while maintaining the confidentiality of their own data. MPC protocols were first proposed in the 1980s (e.g., [11], [5], [1]), and several frameworks for implementing MPC have been developed over the last decade (e.g., [4], [3], [2]). For overviews of MPC and other secure computation techniques see, e.g., [10], [8], [6]. Using MPC, parties $P_1, \ldots, P_n$ having private input data $x_1, \ldots, x_n$ can compute a function $(y_1, \ldots, y_n) = f(x_1, \ldots, x_n)$ so that each party P_i learns only its intended output y_i and nothing more. In the setting of cyber risk assessment, organizations can compute relevant statistics and analyses on the global infrastructure while keeping the details of their local infrastructure and vulnerabilities private. Thus, MPC enables the desired level of collaboration while overcoming the privacy concerns of information sharing.

Two computations with such concerns are selecting IP addresses to blacklist and aggregating the output from vulnerability scanners. IP blacklists must walk the line between being so permissive that they fail to block malicious addresses and retaining so many addresses that enforcing the blacklist becomes unmanageable. Each organization would like to learn which addresses its peers have blacklisted so it can decide which addresses it should add and which older addresses it can remove. Similarly, each organization can better prioritize its security updates by learning which vulnerabilities were common and which patches its peers had applied. Aggregating the output of vulnerability scanners, including the severity and number of vulnerabilities and patch recommendations, would allow organizations to identify issues seen on their peers' infrastructures but not yet on their own and focus their resources on remedying them proactively.

This work is sponsored by the Assistant Secretary of Defense for Research and Engineering (ASDR&E) under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions and recommendations are those of the author and are not necessarily endorsed by the United States Government.

We designed and implemented MPC protocols for the IP blacklist and vulnerability data aggregation applications described above. For the IP blacklist application we implemented a threshold set union protocol that takes a list of private IP addresses from each party and returns a list of the addresses that occur more than some threshold number of times. For the vulnerability data aggregation application we design a protocol that takes as input a set of private key-value pairs from each party and sums values with matching keys. For example, letting keys be vulnerability identifiers and values be the number of vulnerable machines, this protocol computes the total number of machines with each vulnerability. By letting keys be vulnerability severities and values be counts of machines with that vulnerability, this protocol also allows parties to compute the global average severity of their known vulnerabilities.

A number of MPC-specific performance issues arise when building protocols for these applications. Therefore, it is necessary to carefully consider precisely what to compute. For example, the running time of the threshold set union protocol grows significantly with input size. Organizations can improve the running time by choosing to run the protocol on subsets of their full blacklist such as only the oldest addresses or only the newest addresses. For the data aggregation application, the most expensive operations are the comparisons used to determine whether two keys match. Applications that do not require keys to be secret, such as the average vulnerability severity computation described above, will be much faster than applications where the keys must be kept private. Thus, determining the appropriate variant to use depends on the exact functionality desired as well as privacy and performance requirements.

In this talk, we describe the design, implementation, and evaluation of MPC protocols for cooperative cyber risk assessment. We discuss the impact of using different MPC frameworks, input sizes, and privacy requirements on performance. We also discuss tradeoffs between functionality, security, and performance that enable the application of MPC to relevant cyber risk assessment problems in practice.

REFERENCES

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

[2] Dan Bogdanov, Sven Laur, and Jan Willemson. Sharemind: A framework for fast privacy-preserving computations. In *Computer Security - ESORICS 2008, 13th European Symposium on Research in Computer Security, Málaga, Spain, October 6-8, 2008. Proceedings*, pages 192–206, 2008.

[3] Martin Burkhart, Mario Strasser, Dilip Many, and Xenofontas A. Dimitropoulos. SEPIA: privacy-preserving aggregation of multi-domain network events and statistics. In *19th USENIX Security Symposium, Washington, DC, USA, August 11-13, 2010, Proceedings*, pages 223–240, 2010.

[4] Ivan Damgård, Martin Geisler, Mikkel Krøigaard, and Jesper Buus Nielsen. Asynchronous multiparty computation: Theory and implementation. In *Proceedings of the 12th International Conference on Practice and Theory in Public Key Cryptography: PKC '09*, pages 160–179, 2009.

[5] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *STOC*, pages 218–229, 1987.

[6] Ariel Hamlin, Nabil Schear, Emily Shen, Mayank Varia, Sophia Yakoubov, and Arkady Yerukhimovich. Cryptography for big data security. In Fei Hu, editor, *Big Data: Storage, Sharing, and Security*. CRC Press, 2016.

[7] Arjen Lenstra and Tim Voss. Information security risk assessment, aggregation, and mitigation. In *Australasian Conference on Information Security and Privacy*, pages 391–401. Springer Berlin Heidelberg, 2004.

[8] Emily Shen, Mayank Varia, Robert K. Cunningham, and W. Konrad Vesey. Cryptographically secure computation. *IEEE Computer*, 48(4):78–81, 2015.

[9] Fabrizio Smeraldi and Pasquale Malacaria. How to spend it: Optimal investment for cyber security. In *Proceedings of the 1st International Workshop on Agents and CyberSecurity*, ACySE '14, pages 8:1–8:4, 2014.

[10] Sophia Yakoubov, Vijay Gadepally, Nabil Schear, Emily Shen, and Arkady Yerukhimovich. A survey of cryptographic approaches to securing big-data analytics in the cloud. In *High Performance Extreme Computing Conference (HPEC), 2014 IEEE*, pages 1–6, Sept 2014.

[11] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *FOCS*, pages 162–167, 1986.