Asymmetric Private Set Intersection with Applications to Contact Tracing and Private Vertical Federated Machine Learning

Nick Angelou, Ayoub Benaissa, Bogdan Cebere, William Clark, Adam James Hall, Michael A. Hoeh,
Daniel Liu, Pavlos Papadopoulos, Robin Roehm, Robert Sandmann, Phillipp Schoppmann, Tom Titcombe

Abstract

We present a multi-language, cross-platform, open-source library for asymmetric private set intersection (PSI) and PSI-Cardinality (PSI-C). Our protocol combines traditional DDH-based PSI and PSI-C protocols with compression based on Bloom filters to reduce communication in the asymmetric setting. Currently, our library supports C++, C, Go, WebAssembly, JavaScript, Python, and Rust and runs on both traditional hardware (x86) and browser targets. We further apply our library to two use cases: (i) a privacy preserving contact tracing protocol that is compatible with existing approaches, but improves their privacy guarantees, and (ii) privacy-preserving machine learning on vertically partitioned data.

Motivations and Contributions

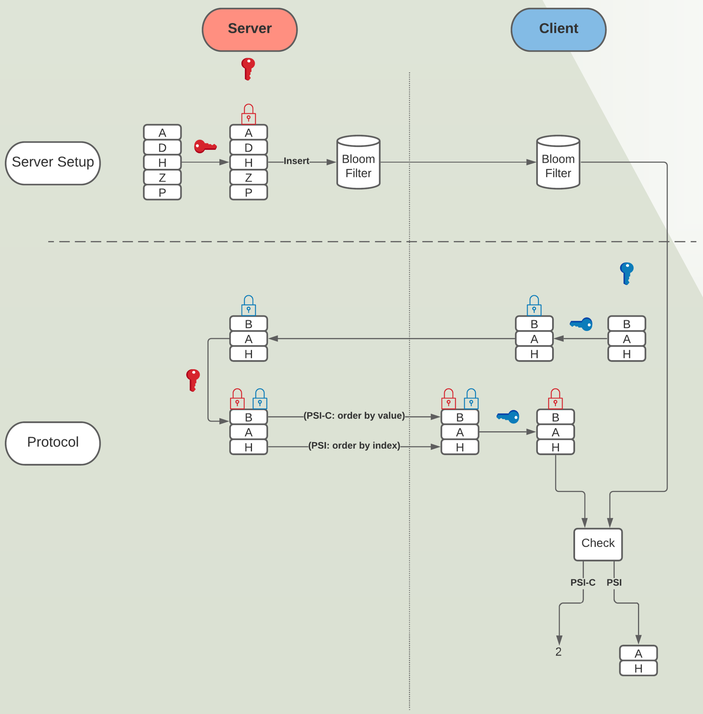
We address the problem of two parties holding two different sets of elements, and they want to compute their intersection (or its size) without having one party leaking its set to the other one. We were motivated by:

- The need for contact tracing applications during the Covid19 pandemic to such technology.
- PSI helps machine learning compute on vertically partitioned data.

We present a versatile open-source library for asymmetric private set intersection (PSI) and PSI-Cardinality (PSI-C). Our library combines the well-studied PSI protocol based on the decisional Diffie-Hellman (DDH) assumption with a Bloom filter compression to reduce the communication complexity. Our library supports bindings for C++, C, Go, JavaScript, Python, and Rust, as well as multiple platforms, including browser targets.

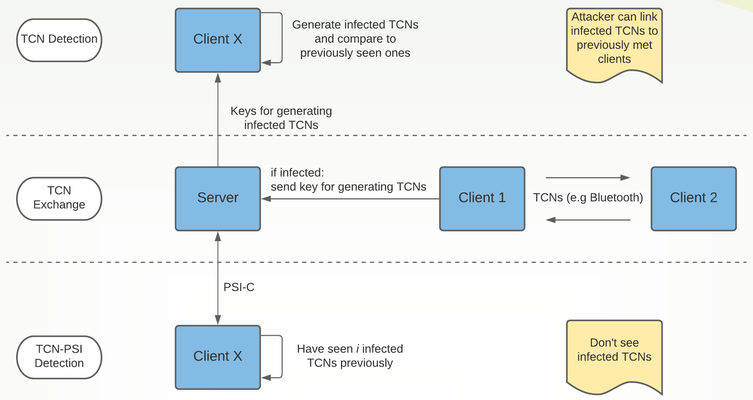
Protocol Description

Both client and server hold a secret key. The client's goal is to learn the encryption of their elements and a compressed representation of the server's elements, all encrypted under the server's key. Our protocol achieves this through a commutative encryption scheme.



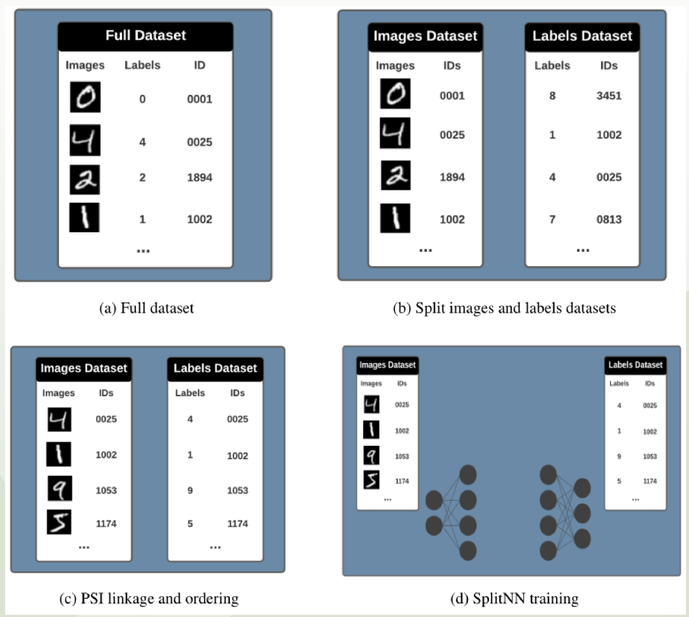
Application 1: Privacy-Preserving Contact Tracing

In the course of the COVID-19 pandemic, several protocols for privacy-preserving contact tracing have been proposed, including DP3T, TCN, and the protocol of Apple and Google. Previous work has shown that these protocols can be susceptible to linkage attacks. We show that these can be mitigated by using PSI-Cardinality, and we implement a version of the TCN protocol that uses our PSI-C library.



Application 2: Private Vertical Federated Machine Learning

In federated learning, a machine learning model is to be trained on data held by multiple parties. In the vertically partitioned setting, the data providers hold different features of a set of overlapping data points. This poses a challenge because the parties have to agree on the data points they have in common. This can be solved using PSI.



(a) Full dataset

(b) Split images and labels datasets

(c) PSI linkage and ordering

(d) SplitNN training

Experimental Evaluation

We compare our different language bindings with each other and with the state-of-the-art implementation for asymmetric PSI by Kales et al. [26]. In our evaluation, we used a server set size of one million, different client set sizes, and a false-positive probability of 10^-9.

[26] Daniel Kales, Christian Rechberger, Thomas Schneider, Matthias Senker, and Christian Weinert. Mobile private contact discovery at scale. In USENIX Security Symposium, pages 1447–1464. USENIX Association, 2019.

Operation	Size	C++	C	Go	Python	WebAssembly	JS	[26]	Comm.	[26] (Comm.)
Server: Setup	$n = 1k$	184.1	181.5	183.9	188.9	1573.8*	9128*	241.54	6.85 MiB	4.19 MiB
	$n = 10k$	184.2	181.7	184.0	188.4	1571.8*	9273*	-	7.42 MiB	-
	$n = 100k$	184.2	181.8	184.3	189.1	1573*	9139.1*	-	7.99 MiB	-
Client: Request	$n = 1k$	0.18	0.17	0.18	0.185	1.57	9.1	2.92†	34.18 KiB	2.00 MiB
	$n = 10k$	1.8	1.77	1.8	1.870	15.6	92	-	341.79 KiB	-
	$n = 100k$	18.09	17.8	18.2	18.2	156.8	909.4	-	3.33 MiB	-
Server: Response	$n = 1k$	0.11	0.11	0.17	0.115	1.2	6.96	†	34.17 KiB	4.07 MiB
	$n = 10k$	1.13	1.13	3.02	1.154	12	70.8	-	341.79 KiB	-
	$n = 100k$	11.3	11.3	29.5	11.5	120.9	701	-	3.33 MiB	-
Client: Intersection	$n = 1k$	0.11	0.11	0.9	0.120	1.2	6.97	†	-	-
	$n = 10k$	1.17	1.16	4.6	1.193	12.1	71.39	-	-	-
	$n = 100k$	11.8	11.6	41.6	11.9	122	710	-	-	-

Conclusion and Future Research

In conclusion, our results show that our library is highly competitive in terms of running time and communication. At the same time, it is flexible enough to support multiple platforms and languages, including browsers. Possible improvements can be made by reducing the setup communication, for example, using Cuckoo filters. Finally, we see extensions like PrivateID as a promising future work.

Acknowledgements

Realizing both the PSI library and the two implementations for contact tracing and vertically partitioned federated learning has only been possible with the help of many people who contributed to this. We extend our deepest gratitude to everybody that has been and continues to be involved in this effort.

Code for our PSI library: https://github.com/OpenMined/PSI
Code for TCN-PSI: https://github.com/bcebere/TCN-PSI
Code for PyVertical: https://github.com/OpenMined/PyVertical
Our paper on Arxiv: https://arxiv.org/abs/2011.09350