UNIVERSITÄT PADERBORN
Die Universität der Informationsgesellschaft

Modelling and Proving Security for a Secure MPC Protocol for Stable Matching

Bachelor's Thesis Proposal
by Ella Vahle

Arbeitsgruppe
Codes und Kryptographie

1 Introduction

Secure multiparty computation (MPC) is a concept for securely computing (potentially probabilistic) functions with multiple parties involved. Each party has private input which must not be disclosed to the other parties. Based on these, the different parties jointly execute a protocol to obtain the function's result. In this context security of the executed protocol typically refers to the privacy of parties' inputs and correctness. To achieve privacy the protocol must ensure that no information about one party's input is leaked to any other parties beyond the function's result. Correctness is attained when the computed result equals the function's output based on the given inputs. To analyse these properties it is important to fix a reasonable attacker model, i.e., what the attacker is capable of. Honest-but-curious (semi-honest) attackers are passive attackers and do not actively interfere with the protocol. They follow instructions correctly and act as expected but try to learn more from the execution than they are supposed to. For a simple example imagine a protocol whose runtime (measured in amount of rounds executed) for input x is longer than for input y. By keeping track of the runtimes an attacker could learn about the other party's input without deviating from the protocol. Malicious attackers on the other hand actively interfere with the protocol and can act differently than advised. It is easy to see that active attackers are more powerful, making active security a stronger requirement than semi-honest security.

Since there are many areas of application where MPC is useful and it is of great theoretical interest, general purpose frameworks were developed early on. With these, arbitrary algorithms can be executed as a secure MPC protocol. To achieve this, the frameworks take as input a description of the algorithm in a specific format, e.g., boolean circuits, and generate a protocol that allows the parties to securely compute the result. After generating the protocol, it is executed.
Even though the idea of these general-purpose frameworks has been around for many decades already and it has been proven that they fulfil the security requirements they are not used as much as one would expect. The main reason for this is that for many

interesting problems general purpose frameworks create solutions that are not efficient enough to be used in practice [HHNZ19, CCD88, GMW87, Yao86]. For these problems special-purpose algorithms with faster execution times need to be constructed. This allows them to be used in practice but involves much more work because security needs to be proven for every single solution separately.

One such problem is finding a stable matching. A matching is a mapping between the members of two disjoint sets. Stable matchings also consider each member's preferences, i.e., a (potentially partial) order over the members of the other set. The matching is stable when there are no two matched pairs so that by switching partners the members get matched to a member of the other set, that they prefer over their current partner. Generally, it is possible to compute matchings where members of one set are matched to multiple members of the other set. The special case of one-to-one matchings is also known as „Stable Marriage", getting its name from the idea of people from two groups being matched to get married.
Such algorithms are used to for example match residents to residency programs [Lü16] or students to public schools [Tul14]. It is worth noting that these are cases where multiple residents/students can be matched to the same member instead of a one-to-one matching. Obviously, the best solution would be that every member is assigned to their first preference, but in reality, this is rarely possible. Stable matchings are a desirable solution in such cases to still achieve a high level of member satisfaction.
One solution for computing a stable one-to-one matching is the Gale-Shapley algorithm, developed by David Gale and Lloyd Shapley, which will also be the one we are interested in.
Ideally, the matching could be computed by a trusted third party to which the parties privately send their inputs. This way, both parties' inputs stay hidden to the other party and are only revealed to the commonly trusted party. Relying on a commonly trusted party has several disadvantages. In many cases it is not even possible to find such a party. But even if it is possible, it can be hard to find one and entail a lot of efforts.
This is exactly an application for secure MPC, i.e., for a protocol which essentially replaces an ideal third party by a secure real-world protocol. Doerner et al. [DEas16] proposed a supposedly secure protocol for this application. Within the protocol they make use of a newly introduced oblivious data structure called *„oblivious linked multi-list"* (OLML) while executing their protocol based on the Gale-Shapley algorithm. The data structure uses data arrays combined with a method by Zahur et al. [ZWR$^+$16] to obliviously permute them. This provides the means to obliviously access data, thus hiding access patterns and the information derivable from these. The authors claim that using the OLML to securely implement the Gale-Shapley algorithm leads to a relatively efficient protocol in the honest-but-curios attacker model. Even though the authors argue why security holds in this setting, it lacks a rigorous formal proof.

2 Current state of research

Stable matching in the context of multiparty-computation comprises multiple different aspects, such as data access strategies, the realized algorithm, or the attacker model, making it an interesting problem both in theory and practice. All of these are research topics of their own with several possible approaches and solutions. Naturally this yields various possibilities of combining them to solve the problem of securely computing a stable matching. As opposed to the discussed approach by Doerner et al. [DEas16] which uses the Gale-Shapley algorithm, Blanton et al. [BSA13] for example use a form of Breadth First Search.

As for the Gale-Shapley algorithm there are several existing approaches.
The usability problem of generic solutions for stable matchings, in particular the Gale-Shapley algorithm, was also mentioned by Golle [Gol06]. To hide the index of the accessed array element, thus making the access appear oblivious, generic solutions produce effort linear in the number of elements. Applying this to array access intense algorithms such as stable matching algorithms results in a lot of overhead. Due to this, he proposed a „practical“ solution with complexity of supposedly $\mathcal{O}(n^3)$ asymmetric cryptographic operations. It is based on matching authorities, and ensured to fulfil privacy and correctness requirements when semi-honesty can be guaranteed for a majority of performing matching authorities.
Problems with Golle's solution, especially a complexity of actually $\mathcal{O}(n^5)$, were later found by Franklin et al. [FGM06], resulting in two new solutions. One is based on Golle's approach and the other one is based on garbled circuits and a protocol by Naor Nissim for secure function evaluation, both with $\mathcal{O}(n^4)$ public key operations and computation complexity respectively.
Previously to Doerner et al. the best approach was proposed by Zahur et al. [ZWR+16]. With a runtime of roughly 33 hours on input of 512 x 512 participants, it is over 40 times slower than Doerner et al.'s protocol.

3 Goals of the thesis

As already mentioned, the solution proposed by Doerner et al. [DEas16] is claimed to be secure but its security has not been formally proven yet. The goal of this thesis is to formalize the rough intuition given by Doerner et al. and provide a formal proof based on the real/ideal world paradigm using the simulation technique introduced by Goldreich [Gol09]. The Simulation Technique was later described by Lindell [Lin17] in an extensive Tutorial which will also be used as a basis for this thesis. The simulation paradigm states the following: We have a functionality modelled as an ideal functionality, i.e., a formal model of the functionality's expected behaviour and a protocol realizing the functionality. If we can simulate what an (honest-but-curious) attacker would see during the execution of the protocol in the real world with real participants, indistinguishably from what they see in the ideal world, the protocol is as secure as computing

the function with the help of a trusted party.
First, we will consider the OLML. According to the simulation technique we will model the notion of an OLML as an ideal functionality. Then we will examine how Doerner et al. realize this functionality in detail, by answering the following questions: How do they transform the input? Which operations are performed on the data? We will then proceed to formally prove security including correctness and privacy. This will be done by showing that it works correctly, meaning that their output equals the ideal model's output and showing that nothing (beyond the function's output) can be learned about the input, respectively. The proof for privacy is based on the simulation paradigm as described above.
After proving security of their proposed OLML implementation we will use it as a black-box when showing the whole protocol's security. To prove the protocol's security, we will carry out the same steps as we did with the OLML, i.e. establishing an ideal model of a stable matching functionality, examining Doerner et al.'s construction and then formally proving security. By showing that the protocol is secure when using the OLML (which was previously proven to be secure) as an ideal blackbox we can conclude that the combination of the algorithm and OLML is also secure, using the well-known composition theorem by Goldreich [Gol09].
Another advantage this modular approach has becomes relevant when the protocol turns out not to be secure in the specified setting. Splitting the protocol into two parts allows us to point out where things go wrong more specifically and adjust the attacker model to potentially show security in a weaker security model or adjust the protocol itself.
It is also possible that the proof turns out to be more straight forward than expected. A reasonable course of action could be to go a step further and investigate how the proven security transfers to other security models, such as an active attacker scenario. Here again, we would be making use of the modular approach, benefiting from its advantages.

4 Preliminary outline of the thesis

1. Introduction
2. Definitions and notation
   a) General
   b) Stable Matching
   c) Secure MPC
3. Prior knowledge
   a) ORAM
   b) Gale-Shapley
4. Oblivious Linked Multi-lists
   a) Ideal model

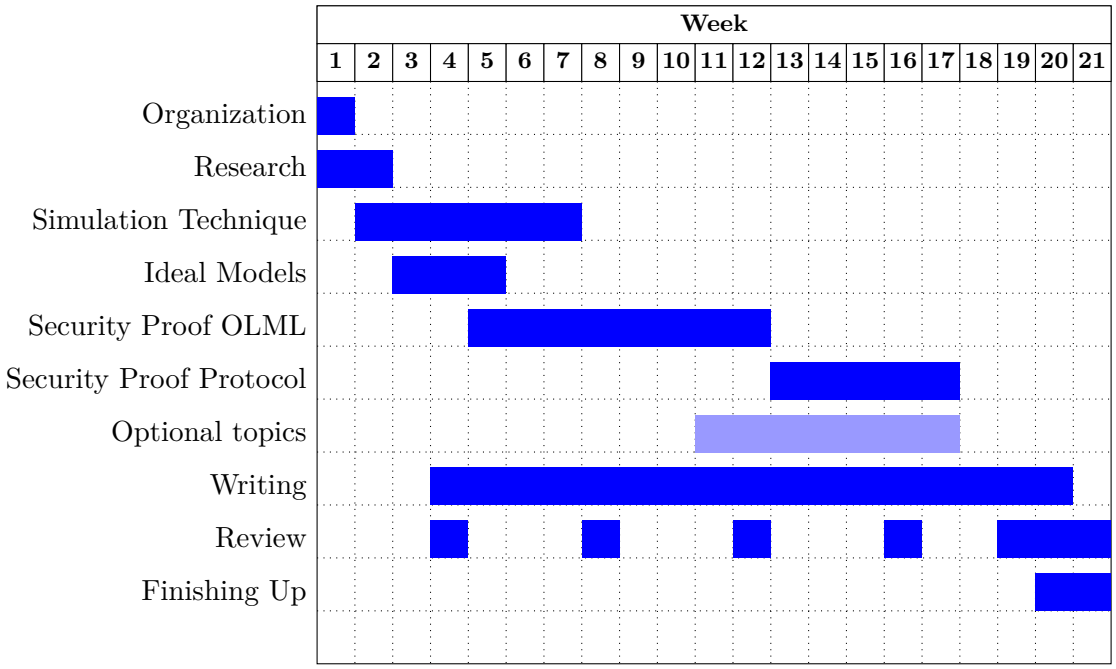
b) Construction

c) Security

d) (optional) Security under different security model (i.e. stronger or weaker depending on the prior results)

5. Applying Oblivious Linked Multi-lists to Gale-Shapley
   a) Ideal model
   b) Construction
   c) Security
   d) (optional) Security under different security model (i.e. stronger or weaker)
6. Conclusion
7. References

5 Work plan



Date:

Prof. Dr. Johannes Blömer

Ella Vahle

References

[BSA13] Marina Blanton, Aaron Steele, and Mehrdad Alisagari. Data-oblivious graph algorithms for secure computation and outsourcing. In *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security*, ASIA CCS '13, page 207–218, New York, NY, USA, 2013. Association for Computing Machinery.

[CCD88] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In Janos Simon, editor, *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 11–19. ACM, 1988.

[DEas16] Jack Doerner, David Evans, and abhi shelat. Secure stable matching at scale. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, page 1602–1613, New York, NY, USA, 2016. Association for Computing Machinery.

[FGM06] Matthew Franklin, Mark Gondree, and Payman Mohassel. Improved efficiency for private stable matching. In Masayuki Abe, editor, *Topics in Cryptology – CT-RSA 2007*, pages 163–177, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred V. Aho, editor, *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 218–229. ACM, 1987.

[Gol06] Philippe Golle. A private stable matching algorithm. In Giovanni Di Crescenzo and Avi Rubin, editors, *Financial Cryptography and Data Security*, pages 65–80, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[Gol09] Oded Goldreich. *Foundations of cryptography: volume 2, basic applications*. Cambridge university press, 2009.

[HHNZ19] Marcella Hastings, Brett Hemenway, Daniel Noble, and Steve Zdancewic. Sok: General purpose compilers for secure multi-party computation. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 1220–1237, 2019.

[Lü16] Ingo Lütkebohle. National Resident Matching Program. 2016 Main Residency Match. `http://www.nrmp.org/wp-content/uploads/2016/04/Main-Match-Results-and-Data-2016.pdf`, 2016. [Online; accessed 04-August-2021].

[Lin17] Yehuda Lindell. *How to Simulate It – A Tutorial on the Simulation Proof Technique*, pages 277–346. Springer International Publishing, Cham, 2017.

[Tul14] Tracy Tullis. How game theory helped improve new york city's high school application process. *The New York Times*, 2:67–79, December 2014.

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th Annual Symposium on Foundations of Computer Science, Toronto, Canada, 27-29 October 1986*, pages 162–167. IEEE Computer Society, 1986.

[ZWR+16] Samee Zahur, Xiao Wang, Mariana Raykova, Adrià Gascón, Jack Doerner, David Evans, and Jonathan Katz. Revisiting square-root oram: Efficient random access in multi-party computation. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 218–234, 2016.