A Framework for Secure Computations With two Non-Colluding Servers and Multiple Clients, Applied to Recommendations

Rummana Firdaus
Department of Computer Science & Engineering
Assistant Professor,GSSSIETW, Mysuru

Abstract-*A generic framework is provided that, with the help of a preprocessing phase that is independent of the inputs of the users, allows an arbitrary number of users to securely outsource a computation to two non-colluding external servers. The approach is shown to be provably secure in an adversarial model where one of the servers may arbitrarily deviate from the protocol specification, as well as employ an arbitrary number of dummy users. These techniques are used to implement a secure recommender system based on collaborative filtering that becomes more secure, and significantly more efficient than previously known implementations of such systems, when the preprocessing efforts are excluded. Different alternatives are suggested for preprocessing, and discuss their merits and demerits.*

Keywords- Secure multi-party computation, malicious model, secret sharing, client-server systems, preprocessing, recommender systems.

I. INTRODUCTION

RECOMMENDATION systems consist of a processor together with a multitude of users, where the processor provides recommendations to requesting users, which are deduced from personal ratings that were initially submitted by all users. It is easy to see that, in a non-cryptographic setup of such a system, the processor is both able to learn all data submitted by the users, and spoof arbitrary, incorrect recommendations. In this work we replace the recommendation processor by a general two-server processor in such a way that, as long as one of the two servers is not controlled by an adversary and behaves correctly, the privacy of the ratings and recommendations of the users is maintained to the fullest extent possible, and a server that is under adversarial control is unable to disrupt the recommendation process in such a way that an incorrectly computed recommendation will not be detected by the requesting user. The result uses a modified version of the standard model for secure multi-party computation, which is a cryptologic paradigm in which the players jointly perform a single secure computation and then abort. In our model the computation is ongoing (recommendations are repeatedly requested) and outsourced to two external servers that do not collude. This approach allows for the involvement of many users that need only be online for very short periods of time in order to provide input data to, or request output data from, the servers. In practice, one of the two servers could be the service provider that wishes to recommend particular services to users, and the other server could be a governmental organisation guarding the privacy protection of users. The role of the second server could also be commercially exploited by a privacy service provider, supporting service providers in protecting the privacy of their customers. While this work focuses on the application of secure recommendation systems, it is pointed out that the underlying framework is sufficiently generic for use in other, similar applications, and also easily extends to model variations involving more than two servers. In general, any configuration is foreseen consisting of multiple servers that have a joint goal of securely delivering a service to a multitude of users. The online phase will be computationally secure as long as one of the servers is honest. However, the application should allow for a significant amount of preprocessing, which is independent of the data, and could be performed during inactive time. Although most related work is only secure in the semi-honest model, security is provided in the malicous model. There are several reasons why security in the malicious model is important. First of all, the main goal of securing the system is that we do not want the servers to learn the personal data of users. We therefore should not just trust them to follow the rules of the protocol. Second, in a malicious model the correctness of the user outputs is better preserved, because outputs cannot be corrupted by one (malicious) server on his own. Third, a malicious server might introduce a couple of new, so called dummy users, which he controls. These dummy users might help him deduce more personal data than is available through the protocol outputs. We show that in our malicious model this is not possible.

A. User-Based Collaborative Filtering

As an example application, a basic implementation for generating recommendations, more specifically a collaborative filtering method with user neighborhood-based rating prediction is used. It might not be the best recommender

system, but merely an introduction to some basic components, and show how they can be implemented securely. During initialisation, each user n uploads to the processor a list of at most M ratings of items, where each rating V(n,m)is represented by a value within a pre-specified interval. Users can update their ratings at any time during the lifetime of the system. A user can, at any time after the initialization, request a recommendation from the processor. When the processor receives such a request from a user, it computes a recommendation for this user as follows. First, it uses the initial S ratings in each list to determine which other users are considered to be similar to the requesting user, i.e. have similarly rated the first S items. The remaining M−S entries in the lists are then used to compute and return a recommendation for the requesting user, consisting of M −S ratings averaged over all similar users.

B. Secure Client-Server Model

The secure framework relies heavily on techniques from the cryptologic area of secure multi-party computation. The problem of secure multi-party computation considers a fully connected communication network of n parties, where the parties wish to compute the outcome of a given function f on their respective inputs. However, apart from the output, no information on the inputs should leak during the course of the computation. In this work we consider functions f that correspond with the computation of a recommendation for a user. The ideal solution to the problem of secure multi-party computation involves an independent, incorruptible mediator that privately takes all the required inputs, computes and reveals the desired output to the appropriate parties, and then forgets everything. The research area of secure multiparty computation studies techniques that allow the parties to simulate the behavior of such a mediator.These simulations have the following structure. First, there is an input phase that enables the parties to encrypt their respective inputs for use in the secure computation. Then a computation phase takes place during which an encrypted output of the function f is computed from the encrypted inputs. Finally, an output phase takes place where the output is decrypted, and sent to the appropriate parties. Consequently, this data is completely independent of the input data of the parties. The goal of the preprocessing is to remove as much of the complexity and interaction from the actual computation as possible, which as a result makes this computation extremely efficient.

C. Adversarial Behavior and Security

Consider an adversary that is able to take full control of one of the two servers involved in the two-party computation with preprocessing. Moreover, the adversary can initially introduce an arbitrary number of so-called dummy users into the system, which are also under his complete control. Dummy users can in particular, like ordinary users, input or update their ratings, and request recommendations. The adversary can read all data available to the entities that are under his control, and can make them deviate from the cryptographic protocol specification arbitrarily. The goal is to prove that the actions of the adversary have essentially no impact on the outcome or the security of the protocol. In order to describe the security level of our secure recommendation system we make use of the real/ideal world paradigm. This paradigm is based on the comparison between two scenarios. In the first scenario, the real world, the adversary participates in a normal protocol execution. In the second scenario, the ideal world, all participants in the protocol have black-box access to the functionality that the protocol in the real world attempts to emulate. Moreover, there exists a simulator that creates a virtual environment around the adversary, and interacts with the adversary in a special fashion. The goal of the simulator in the ideal world is to simulate the expected world-view of the adversary during a real protocol execution. The security analysis now consists of showing that the adversary cannot distinguish whether it is acting during a protocol execution in the real world, or in the ideal world .Since no effective attack is possible in the ideal world, it then follows that no effective attack is possible in the real world either.

D. Secure Recommendations

Although this framework allows clients to download any value from the processor, we focus on our recommender application. After a recommendation has been computed, the outputs have to be sent to the requesting user. Since we want the user to be able to verify the correctness of the output, but we do not want to reveal the private authentication keys to him, we need an additional protocol. Although users can be untrusted, they assume a secure blackboard, where users can upload values, which can be read by other users but not changed. It is assumed that the existence of at least one honest server. A user requesting a recommendation will learn the amount of similar users. The reason is that this approach avoids having to perform a secure integer division protocol which is quite expensive. That threat is mitigated by implementing a secure integer division protocol. The most important difference is that their protocol is secure in the semi-honest model, while ours is secure in the malicious model. In the recommender system, the user is moreover assured that his outputs, the predicted ratings, are correctly computed.

II. PROPOSED METHOD

In this work the recommendation processor is replaced by a general two-server processor in such a way that, as long as one of the two servers is not controlled by an adversary and behaves correctly The privacy of the ratings and recommendations of the users is maintained to the fullest extent possible, and a server that is under adversarial control is unable to disrupt the recommendation process in such a way that an incorrectly computed recommendation will not be detected by the requesting user. In this model the computation is ongoing (recommendations are repeatedly requested) and outsourced to two external servers that do not collude. This approach allows for the involvement of many users that need only be online for very short periods of time in order to provide input data to, or request output data from, the servers. In practice, one of the two servers could be the service provider that wishes to recommend particular services to users, and the other server could be a governmental organization guarding the privacy protection of users. The role of the second server could also be commercially exploited by a privacy service provider, supporting service providers in protecting the privacy of their customers. The SPDZ framework, is used which enables secure multi-party computations in the malicious model, extended it to the client-server model, and worked out a secure recommendation system within this setting. Not only did this lead to a recommendation system that is secure in the malicious model, but also the online phase became very efficient in terms of computation and communication complexity. To securely compute a recommendation within SPDZ, we had to develop a secure comparison protocol and a secure integer division protocol.

Advantages of Proposed System:

The result uses a modified version of the standard model for secure multi-party computation, which is a cryptologic paradigm in which the players jointly perform a single secure computation and then abort. Although most related work is only secure in the semi-honest model, security is provided in the malicious model.

III. CONCLUSIONS

A general framework is provided for outsourcing ongoing computations, which is suitable for dealing with a large number of users, and is provably secure in the malicious model. This framework is then applied to the problem of secure recommendation and, given a sufficient amount of pre-computed data, leads to extremely efficient implementations.

The approach is very generic and easily allows for variations. Although the case of two servers is considered, the model described is easily extended to an arbitrary number of servers. As a consequence, many other secure applications are possible using our framework.

REFERENCES

[1] R. Bendlin, I. Damgård, C. Orlandi, and S. Zakarias, "Semi- homomorphic encryption and multiparty computation," in Advances in Cryptology (Lecture Notes in Computer Science), vol. 6632. Berlin, Germany: Springer-Verlag, 2011, pp. 169–188..

[2] S. Zhang, J. Ford, and F. Makedon, "Deriving private information from randomly perturbed ratings," in Proc. 6th SIAM Int. Conf. Data Mining, 2006, pp. 59–69.

[3] Z. Erkin, T. Veugen, T. Toft, and R. L. Lagendijk, "Generating pri- vate recommendations efficiently using homomorphic encryption and data packing," IEEE Trans. Inf. Forensics Security, vol. 7, no. 3, pp. 1053–1066, Jun. 2012.

[4] A. Peter, E. Tews, and S. Katzenbeisser, "Efficiently outsourcing mul- tiparty computation under multiple keys," IEEE Trans. Inf. Forensics Security, vol. 8, no. 12, pp. 2046–2058, Dec. 2013.

[5] I. Damgård, M. Keller, E. Larraia, V. Pastro, P. Scholl, and N. P. Smart, "Practical covertly secure MPC for dishonest majority—Or: Breaking the SPDZ limits," in Computer Security (Lecture Notes in Computer Science), vol. 8134. Berlin, Germany: Springer-Verlag, 2013.

[6] Y. Huang, J. Katz, and D. Evans, "Efficient secure two-party com- putation using symmetric cut-and-choose," in Advances in Cryptology (Lecture Notes in Computer Science), vol. 8043. Berlin, Germany: Springer-Verlag, 2013, pp. 18–35.

[7] R. Canetti, "Universally composable security: A new paradigm for cryptographic protocols," in Proc. 42nd IEEE Symp. Found. Comput. Sci., Oct. 2001, pp. 136–145.

[8] A. C. Yao, "Protocols for secure computations," in Proc. IEEE 54th Annu. Symp. Found. Comput. Sci. (FOCS), Nov. 1982, pp. 160–164.

[9] Y. Lindell, "Fast cut-and-choose based protocols for malicious and covert adversaries," in Advances in Cryptology (Lecture Notes in Computer Science), vol. 8043. Berlin, Germany: Springer-Verlag, 2013, pp. 1–17.

[10] M. Atallah, M. Bykova, J. Li, K. Frikken, and M. Topkara, "Private collaborative forecasting and benchmarking," in Proc. ACM Workshop Privacy Electron. Soc. (WPES), 2004, pp. 103–114.