A Practical k-Anonymous Recommender System

Athanasios Zigomitros, Achilleas Papageorgiou, Constantinos Patsakis
Department of Informatics
University of Piraeus
{azigomit,achipapa,kpatsak}@unipi.gr

***Abstract*—Providing users with a lot of information might sound ideal in many scenarios, nonetheless, this may often be very annoying for the end user. To limit the amount of information that has to be processed by the user, to a set that is more relevant for his needs, most service providers use recommender systems. Undoubtedly, to provide someone with proper recommendations, one needs some background knowledge about this individual. Nonetheless, depending on the nature of the query, the database and the means to acquire this background knowledge, a lot of sensitive user information can be leaked to the system, something that many companies try to monetize. To address such privacy issues, privacy preserving recommendation systems have been developed. In this work we introduce an efficient collaborative filtering system which uses state of the art encryption primitives to offer privacy to both the server and the client, while simultaneously providing quality recommendations to the latter.**

***Index Terms*—Privacy, Recommender Systems, Secure Multi-party Computation**

I. Introduction

We live in an era where the information flows in huge quantities from and towards users. While the latter can be considered a tremendous achievement, in many cases it can be quite troublesome for the end-user. A typical such example is the case of a user who wants to buy an item from an e-shop and the wide variety of products can overwhelm the user so much that he cannot choose what he wants. To facilitate such choices, recommender systems were introduced. One of the most well-known techniques is the use of collaborative filtering, where users share their ratings for some items with a trusted authority or in a distributed way. This way, users are grouped in clusters with similar preferences and the system can easily provide a recommendation to a new user when he requests it by categorizing him to one of these clusters.

While the scenario might seem benign, the privacy of the user can be significantly exposed if his preferences are linked with other data or even with the context of his choices. For instance, if the e-shop is selling books one can deduce political and religious beliefs or even sexual preferences. Clearly, depending on the context of the underlying database, the amount of information as well as the quality that can be extracted for individual users can be significant and quite sensitive. In fact, many companies put a lot of efforts in user profiling as the analytics that they can produce from fine-grained profiles can be considered invaluable.

One of the main approaches in recommendation systems is Collaborative Filtering (CF) in which the underlying idea is that users would prefer recommendations of like-minded users. In this setting, one can consider that the recommendation system has a $\mathcal{P} \times \mathcal{U}$ matrix $\mathcal{M}$ which contains the ratings of $\mathcal{U}$ users for $\mathcal{P}$ products. Apparently, the matrix can be considered sparse as most users will provide input for items they clearly suggest or detest, which most of the times is a small subset of all the items. Therefore, deriving the best recommendation for a new user is equivalent to finding the maximum inner product of the ratings of the new user with the rows of matrix $\mathcal{M}$. To speed up this process, recommendation systems tend to group individuals into clusters and try to determine in which cluster an individual belongs to. This way, the inner products that have to be calculated are significantly reduced.

To address privacy issues in CF there are several approaches, none of which comes without a cost. For instance, some solutions may rely on trusted third parties, in other cases the recommendations contain a lot of noise so they do not meet users' expectations in terms of quality, or the computational and bandwidth overhead undermine significantly the applicability of the proposed solution. Following the line of thought of Weng et al. [1], we extend their work to provide more privacy for both the server as well as the client and indicate how the user can retrieve better results from these recommendations.

II. Related work

A. Privacy Preserving Collaborative filtering

One of the first approaches in privacy preserving collaborative filtering was from Polat and Du [2]. Their approach was quite straight forward, in their scheme a central server stores the ratings of all users as in the original CF scenario, nonetheless, no user submits his true ratings. When submitting a rating, each user adds Gaussian noise in his measurements. Therefore, while his actual records are not disclosed, the added noise is cancelled out when the server aggregates the measurements for all users. In other cases, such as Shokri et al [3], users are divided in clusters of similar ratings so their ratings are mixed in a way that the records have some sort of similarity, yet the actual values are not disclosed. Other centralized approaches may use singular value decomposition [4] or microaggregation [5], [6]. Beyond centralised methods, there are several solutions in the decentralised setting such as [7], [8], [9], [10]. For a more thorough overview of privacy preserving collaborative filtering the interested reader may refer to [11].

While the aforementioned methods may provide privacy to the users whose ratings have been stored in the recommendation database, few schemes guarantee the privacy of the

users performing the query to the service provider and far less manage to do it without the use of P2P. Therefore, when the underlying data are sensitive; e.g. consider the case of a health recommender scheme in a smart city [12], [13], the privacy of individuals is of paramount importance.

Of specific interest is the recent scheme of Weng et al. [1] which exploits the properties of locality-sensitive hashing (LSH) [14] to hide the user's input. This way, similar data streams are mapped to similar hashes with overwhelming probability. While LSH can efficiently hide a lot of information, hardly can anyone consider them as actual secure instances. From their very definition one can understand that if a service provider is given the hash value, he may not recover the real input, LSH is a one-way function, nevertheless, he can easily categorize him in his clusters, undermining the actual goal of the scheme.

B. NTRU

NTRU [15] is a secure lattice-based public key encryption algorithm and can be considered as one of candidates for the post-quantum era, as its security is expected to be unaffected by the introduction of quantum computers. Despite its security, the algorithm is so efficient that it can be even compared with symmetric ciphers [16] and has many algebraic properties. The algorithm is *somewhat homomorphic* as it manages to transfer, for a limited amount of operations, both addition as well as multiplication.

NTRU works as follows. Firstly, we select the security parameters of the algorithm, namely N, p and q, Table I indicates the offered security for some selections according to X9.98 standard. In practice, N is a prime number, used to determine the degree of the polynomials that are going to be use as every polynomial is reduced modulo $x^N - 1$. Moreover, in NTRU we use two moduli numbers a "large" (q) and a "small" one (p), which are set to 2048 and 3 respectively. All operations are performed over $\mathbb{Z}_q[x]/(x^N-1)$ and $\mathbb{Z}_p[x]/(x^N - 1)$.

To instantiate the algorithm, we first select two random polynomials f and g with their coefficients being only -1, 0 and 1. For polynomial f there is an additional requirement to be invertible both in $\mathbb{Z}_q[x]/(x^N - 1)$ and $\mathbb{Z}_p[x]/(x^N - 1)$. These two inverses are denoted f_q and f_p respectively. The public key h is defined as $h = pgf_q$ and (f, f_p) is the private key. To encrypt a message m, we map m to a polynomial with coefficients only being -1, 0 and 1 and pick a random noise polynomial r, again with coefficients of -1, 0 and 1. The encrypted message is then:

$$c = hr + m \in \mathbb{Z}_q[x]/(x^N - 1)$$

To decrypt ciphertext c, the recipient multiplies it with f and rearranges the coefficients to reside within $[-q/2, q/2]$ and reduces it modulo p. Finally, we multiply the result with f_p.

Contrary to many algorithms, NTRU decryption can fail, nonetheless, this probability can be easily bounded to practical limits so that the implementations do not face such issues. To this end, the amount of 1s, 0s and -1s of f, g, m and r need to be specific. In practice, a message can be decrypted only if the following inequality holds:

$$\|f * m + p * r * g\|_\infty \leq q$$

otherwise the result will be a random polynomial.

Security Level	RSA	Elliptic Curves	NTRU p	NTRU q	NTRU n	NTRU Public key (bits)
128	3072	256	3	2048	439	4829
192	7680	384	3	2048	593	6523
256	15360	521	3	2048	743	8173

TABLE I
PARAMETERS FOR THE MOST POPULAR SECURITY LEVELS (IN BITS). FOR RSA AND ELLIPTIC CURVES, THE NUMBERS DENOTE THE LENGTH (IN BITS) OF THE UNDERLYING MODULO FIELD. FOR NTRU, THE NUMBERS ARE PRECISE AND RECOMMENDED BY SECURITYINNOVATION [17].

III. PROPOSED SCHEME

In our scheme, we assume that the server has a database which contains the rankings of $\mathcal{U}$ users on $\mathcal{P}$ items. To generate this database, the service provider has devoted several efforts and resources [18], [19], hence he does not want it to share it with others. Nonetheless, for the sake of providing his service, he chooses to selectively share distorted rows of his dataset.

Our proposed scheme, illustrated in Figure 1, consists of two layers. The first layer provides the user with a snapshot of the contents of the database, so that he can select the rows which match best to his query. Nonetheless, the snapshot that the user has each time is different and leaks the least possible information about the database. Finally the second layer, provides the user with the responses that he needs in a k-anonymous setting.

More precisely, our proposed scheme consists of the following steps. First the user encodes his ratings in a vector that he hashes using a local-sensitive hash function H deriving LSH_C. Then, using his public key h, he uses NTRU to encrypt it, and sends $c = hr + LSH_C$ to the service provider. Upon receiving the message, the server exploits the additive homomorphic property of NTRU and subtracts it from each of his encryptions of the stored hashed recommendations. Therefore, the servers computes:

$$\begin{aligned} c_i &= hr_i + LSH_{S_i} - hr + LSH_C + e_i \\ &= h\rho + \Delta_{LSH_i} + e_i \end{aligned}$$

where e_i is the added noise, $\rho = r_i - r$ and Δ_{LSH_i} denotes the difference between the LSH of the client rankings and the LSH of the rankings of row i in the server. The use of LSH reduces significantly the dimensionality of the matrix, as such tables may have thousands of columns and follows the schema of Weng et al [1]. Moreover, the LSH values of each record can be cached to improve the performance of the scheme. Nonetheless, since LSH are not cryptographically secure, an adversary, after several queries could potentially derive more information about the actual values of the rows. Therefore, a random noise is appended each time so that the derived

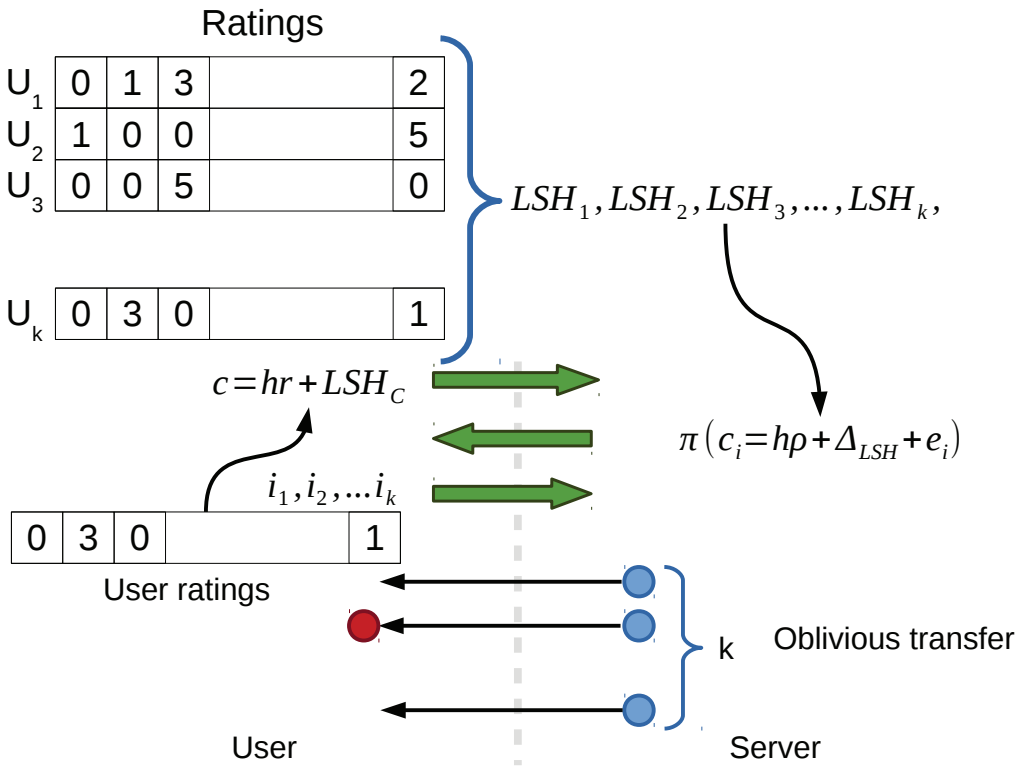

Fig. 1. The proposed scheme.

snapshot is different and does not leak any information. Then, our recommender system permutes the results using a keyed pseudo-random permutation $\pi_x : \{0,1\}^k \rightarrow \{0,1\}^k$ and sends them to the user.

On receiving the results, the user can decrypt them to find which rows are closer to his submitted profile. Note that since in the previous step the records where permuted and a random noise was added, in each execution, a different snapshot is received. Hence, even if the user submits the same query, and the database has not changed, he cannot be not sure whether he has selected the same record. Based on the similarity that he finds, the user selects his record of choice and more $k-1$ random records and submits their indexes randomly arranged, recording the position of his desired index.

Finally, the server and the user perform an oblivious transfer protocol so that the user gets the desired record, without disclosing which one it is, or the server leaking any information for the rest of his rows. For this task, the simple protocol of Chou and Orlandi [20] can be used, which in our scenario would require $(k+1)\mathcal{P}+2$ exponentiations and the exchange of $2k\mathcal{P}$ group elements. In this execution of the oblivious transfer protocol, the server provides the distorted version of the k records that the user requested in the same order the that the user provided them, and the user, in each round, selects the selected index. After this step, the client only learns the distorted values of one row of the matrix which are very close to his ratings.

IV. Discussion

The security of the proposed scheme relies on several cryptographic primitives. Firstly, the privacy of the user is guaranteed by the security of the NTRU algorithm and afterwards by the security of the oblivious transfer protocol. On the other hand, the security of the service provider; the fact that his records are not leaked is guaranteed by the non-invertness of LSH as well as the added noise and the security of the oblivious transfer protocol. All the above, are considered well established cryptographic primitives so the proposed scheme can be considered secure.

The communication cost of the proposed scheme can be characterised tolerable. More precisely, if we select the well-known 1M MovieLens dataset[1], containing the ratings of 6040 users on 3952 movies, using security of 128 bits and 20-anonymity, the communication cost can be broken down as follows. Initially, the client sends a message which is one NTRU encryption, so the cost is 604B. The server then responds with 6040 NTRU encryptions, which account for approximately 3.48 MB, to which the user responds with 20 indexes, which are 75B. Finally, for the oblivious transfer, 2*20*3952=158080 group elements must be exchanged with the Chou and Orlandi protocol. For the 128-bit security setting, an elliptic curve group element can be described with 512 bits, or 257 if compression is enabled. Therefore, the communication overhead for the oblivious transfer is 4.8MB,

[1]http://grouplens.org/datasets/movielens/1m/

raising the total communication overhead to approximately 8.3MB.

V. Conclusions

The continuous collection of user data coupled with the large quantities of information in which a user has to derive his data are pushing towards the development of privacy preserving recommendation systems. In many instances, the queries may disclose a lot of private user information, therefore, in our approach, not only the server database has to be kept private or the preferences of the users whose records are stored, but additionally, the user who asks for the recommendation has to keep his query private. To this end, we propose an efficient yet private protocol which provides accurate recommendations to the user at a tolerable communication and computational cost. In the future, we plan to provide formal proofs of the security of the scheme as well as a practical evaluation of the scheme for the accuracy of the provided recommendations.

Acknowledgments

This work was supported by the European Commission under the Horizon 2020 Programme (H2020), as part of the *OPERANDO* project (Grant Agreements no. 653704).

The publication of this paper has been partly supported by the University of Piraeus Research Center.

References

[1] L. Weng, L. Amsaleg, A. Morton, and S. Marchand-Maillet, "A privacy-preserving framework for large-scale content-based information retrieval," *Information Forensics and Security, IEEE Transactions on*, vol. 10, no. 1, pp. 152–167, 2015.

[2] H. Polat and W. Du, "Privacy-preserving collaborative filtering using randomized perturbation techniques," in *null*. IEEE, 2003, p. 625.

[3] R. Shokri, P. Pedarsani, G. Theodorakopoulos, and J.-P. Hubaux, "Preserving privacy in collaborative filtering through distributed aggregation of offline profiles," in *Proceedings of the third ACM conference on Recommender systems*. ACM, 2009, pp. 157–164.

[4] H. Polat and W. Du, "Svd-based collaborative filtering with privacy," in *Proceedings of the 2005 ACM symposium on Applied computing*. ACM, 2005, pp. 791–795.

[5] F. Casino, J. Domingo-Ferrer, C. Patsakis, D. Puig, and A. Solanas, "Privacy preserving collaborative filtering with k-anonymity through microaggregation," in *e-Business Engineering (ICEBE), 2013 IEEE 10th International Conference on*. IEEE, 2013, pp. 490–497.

[6] ——, "A k-anonymous approach to privacy preserving collaborative filtering," *Journal of Computer and System Sciences*, vol. 81, no. 6, pp. 1000–1011, 2015.

[7] C. Kaleli and H. Polat, "P2p collaborative filtering with privacy," *Turkish Journal of Electrical Engineering & Computer Sciences*, vol. 18, no. 1, pp. 101–116, 2010.

[8] I. Yakut and H. Polat, "Arbitrarily distributed data-based recommendations with privacy," *Data & Knowledge Engineering*, vol. 72, pp. 239–256, 2012.

[9] ——, "Estimating nbc-based recommendations on arbitrarily partitioned data with privacy," *Knowledge-Based Systems*, vol. 36, pp. 353–362, 2012.

[10] S. Shang, Y. Hui, P. Hui, P. Cuff, and S. Kulkarni, "Beyond personalization and anonymity: towards a group-based recommender system," in *Proceedings of the 29th Annual ACM Symposium on Applied Computing*. ACM, 2014, pp. 266–273.

[11] F. Casino, C. Patsakis, D. Puig, and A. Solanas, "On privacy preserving collaborative filtering: Current trends, open problems, and new issues," in *e-Business Engineering (ICEBE), 2013 IEEE 10th International Conference on*. IEEE, 2013, pp. 244–249.

[12] A. Solanas, C. Patsakis, M. Conti, I. Vlachos, V. Ramos, F. Falcone, O. Postolache, P. Perez-martinez, R. Pietro, D. Perrea *et al.*, "Smart health: a context-aware health paradigm within smart cities," *Communications Magazine, IEEE*, vol. 52, no. 8, pp. 74–81, 2014.

[13] F. Casino, E. Batista, C. Patsakis, and A. Solanas, "Context-aware recommender for smart health," in *Smart Cities Conference (ISC2), 2015 IEEE First International*. IEEE, 2015, pp. 1–2.

[14] A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," in *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*. IEEE, 2006, pp. 459–468.

[15] J. Hoffstein, J. Pipher, and J. H. Silverman, "Ntru: A ring-based public key cryptosystem," in *Algorithmic number theory*. Springer, 1998, pp. 267–288.

[16] J. Hermans, F. Vercauteren, and B. Preneel, "Speed records for ntru," in *Topics in Cryptology-CT-RSA 2010*. Springer, 2010, pp. 73–88.

[17] J. Hoffstein, J. Pipher, J. M. Schanck, J. H. Silverman, W. Whyte, and Z. Zhang, "Choosing parameters for ntruencrypt," Cryptology ePrint Archive, Report 2015/708, 2015, http://eprint.iacr.org/.

[18] A. Westin, "Freebies and privacy: What net users think," technical report, opinion research corporation, Tech. Rep., 1999.

[19] C. Patsakis and A. Solanas, "Privacy as a product: A case study in the m-health sector," in *Information, Intelligence, Systems and Applications (IISA), 2013 Fourth International Conference on*. IEEE, 2013, pp. 1–6.

[20] T. Chou and C. Orlandi, "The simplest protocol for oblivious transfer," in *Progress in Cryptology–LATINCRYPT 2015*. Springer, 2015, pp. 40–58.