Privacy Aware Recommender Service using Multi-agent Middleware – An IPTV Network Scenario

Ahmed M. Elmisery and Dmitri Botvich
Telecommunications Software & Systems Group-TSSG
Waterford Institute of Technology-WIT, Co. Waterford, Ireland
E-mail: ahmedmohmed2001@gmail.com

Keywords: clustering, IPTV network, recommendation-services, multi-agent, secure multiparty computation

Received: October 28, 2011

IPTV service providers are starting to realize the significant value of recommender services in attracting and satisfying customers as they offer added values e.g. by delivering suitable personalized contents according to customers personal interests in a seamless way, increase content sales and gain competitive advantage over other competitors. However the current implementations of recommender services are mostly centralized combined with collecting data from multiple users that cover personal preferences about different contents they watched or purchased. These profiles are stored at third-party providers that might be operating under different legal jurisdictions related to data privacy laws rather than the ones applied where the service is consumed. From privacy perspective, so far they are all based on either a trusted third party model or on some generalization model. In this work, we address the issue of maintaining users' privacy when using third-party recommender services and introduce a framework for Private Recommender Service (PRS) based on Enhanced Middleware for Collaborative Privacy (EMCP) running at user side. In our framework, PRS uses platform for privacy preferences (P3P) policies for specifying their data usage practices. While EMCP allows the users to use P3P policies exchange language (APPEL) for specifying their privacy preferences for the data extracted from their profiles. Moreover, EMCP executes a two-stage concealment process on the extracted data which utilize trust mechanism to augment the recommendation's accuracy and privacy. In such case, the users have a complete control over the privacy level of their profiles and they can submit their preferences in an obfuscated form without revealing any information about their data, the further computation of recommendation proceeds over the obfuscated data using secure multi-party computation protocol. We also provide an IPTV network scenario and experimentation results. Our results and analysis shows that our two-stage concealment process not only protect the users' privacy, but also can maintain the recommendation accuracy.

Povzetek: Članek obravnava priporočanje vsebin uporabnikom televizije IP, ki spoštuje uporabnikovo zasebnost.

1 Introduction

Internet Protocol Television (IPTV) is a video service providing IP broadcasts and video on demand (VOD) over a broadband IP content delivery network (CDN) specialized in video services. The IPTV user has access to myriads of video content spanning IP Broadcast and VOD [1]. In this context, it is difficult for them to find content that matches their preferences from the huge amount of video content available. In order to attract and satisfy these users, IPTV service providers employ recommendation services to increase their revenues and offer added value to their patrons. In the same time, Recommender services can improve the overall performance of the current IPTV network by building up an overlay to increase content availability, prioritization and distribution.

Recommender service offers referrals to users by building up users' profiles (explicit or implicit) based on their past ratings, behaviour, purchase history or demographic information. In the context of this work, a profile is a list comprises the video contents the user has watched or purchased combined with their meta-data extracted from the content provider (i.e. genres, directors, actors and so on) and the ratings the user gave to these contents. Maintaining the quality of offered referrals and quickly react to problems raised from merging data from different sources requires a lot of expertise, and not all IPTV providers have the ability to construct and interpret recommendation models. Therefore, there is a market for specialized firms on users' profiles storage and analysis. But there are some challenges face this business model, such as security and privacy. Because collected data from users cover personal information about different contents they have watched or purchased and these profiles might be stored at third-party providers that might be operating under different legal jurisdictions related to data privacy laws rather than the ones applied where the service is consumed. This is a serious threat to individual privacy since this data can be used for unsolicited marketing, government surveillance, profiling of users, misused, furthermore it can be sold by providers when they face bankruptcy. As a matter of fact, users are more likely willing to give more truthful preferences if privacy measurements are provided or if they assured that the data does not leave their personal devices until it is properly desensitised.

The organization for economic co-operation and development (OECD) [2] formulated sets of principles for fair information practice that can be considered as the base for privacy laws. These principles allow the users to control the data they provide for recommender services operating at remote sites, they can be described as follows:

1. Collection limitation: data collection and usage for a recommender service should be limited only to the data it requires to offer appropriate service.
2. Data quality: data should be used only for the relevant purposes for which it is collected.
3. Purpose specification: data controllers should specify up front how they are going to use data and users should be notified up front when a system will use it for any other purpose.
4. Use limitation: data should not be used for purposes other than those disclosed under the purpose specification principle without user consent.
5. Security safeguards: data should be protected with reasonable security safeguards (encryption, secure transmission channels, etc).
6. Openness: the user should be up front notified when the data collection and usage practices started away.
7. Individual participation: users should have the right to insert, update, and erase data in their profiles stored at data controllers.
8. Accountability: data controllers are responsible for complying with the principles mentioned above.

In this work, we present an enhanced middleware for collaborative privacy (*EMCP*) that allows creating reasonable referrals without breaching user privacy. *EMCP* employs a set of mechanisms to allow users to share their data among each other in the network to form a group to attain collaborative privacy. The users' cooperation is needed not only to protect their privacy but also to allow the service to run properly. Highly reputable peers aggregate participants' preferences then encrypt these collected profiles using homomorphic encryption in order to permit particular operations to be performed on encrypted data without need for prior decryption then they submit these profiles to PRS in order to produce referrals. The encrypted profiles hide the identities of participants, and thus hamper the ability for the untrusted PRS to invade users' privacy by profiling or tracking them. However, participants cannot trust each other as well and hence the aggregation process should not expose their preferences. Hence, we proposed a trust based obfuscation mechanism, which designed especially to obfuscate items' ratings before their submission to these highly reputable peers.

This approach preserves the aggregates in the dataset to maximize the usability of information in order to accurately predicate ratings for items that have not consumed before by the group members. In addition, *EMCP* employs interpersonal trust between users to enhance recommendation accuracy and preserve privacy. The enhancement in accuracy is achieved by employing trust based heuristics to propagate and spot users whom are trustworthy with respect to the target user. Moreover, trust based heuristics enhance privacy by transforming participants' data in different ways based on different trust levels to hide the raw ratings. Thus, In contrast to a single obfuscation level scenario where only one obfuscated copy is released for all users using fixed parameters for the obfuscation mechanism, now multiple differently obfuscated copies of the same data is released for different requests with different trust levels. The more trusted the target-user is the less obfuscated copy he can access. These different copies can be generated in various fashions. They can be jointly generated at different times upon receiving new request from target user, or on demand fashion. The latter case gives users maximum flexibility.

In rest of this work, we will generically refer to news programs, movies and video on demand contents as Items. This paper is organised as follows. In Section 2, related works are described. Section 3 presents the threat model assumed in this work .Section 4 introduces IPTV network scenario landing our private recommender service. The proposed solution based on *EMCP* is introduced in Section 5. In Section 6, the two-stage concealment process is described in details. Proof of security and correctness for the two-stage concealment process is demonstrated in Section 7. In Section 8, the Results from some experiments on the proposed mechanisms are reported. Finally, the conclusions and recommendations for future work are given in Section 9.

2 Related Works

The majority of the existing recommender services are based on collaborative filtering techniques that build users' profiles in two ways on ratings (explicit rating procedures) or on log archives (implicit rating procedures) [3]. These procedures lead to two different approaches for to collaborative filtering including the rating based approaches and log based approaches. The majority of the literature addresses the problem of privacy on collaborative filtering techniques, due to it being a potential source of leakage of private information shared by the users as shown in [4]. In [5] a theoretical framework is proposed to preserve the privacy of customers and the commercial interests of merchants. Their system is a hybrid recommender system that uses secure two party protocols and public key infrastructure to achieve the desired goals. In [6, 7] a privacy preserving approach is proposed based on peer to peer techniques using users' communities, where the community will have a aggregate user profile representing the group as a whole but not individual users. Personal information will be encrypted and communication will be between individual users but not servers. Thus, the recommendations will be generated on the client side. In [8, 9] another method is suggested for privacy preserving on centralized recommender systems by adding uncertainty to the data by using a randomized perturbation technique while attempting to make sure that the necessary statistical aggregates such as the mean do not greatly get disturbed. Hence, the server has no knowledge about the true values of the individual items' ratings for each user. They demonstrate that this method does not essentially decrease the accuracy obtained in the results. But recent research work [10, 11] pointed out that these techniques do not provide levels of privacy as was previously thought. In [11] it is pointed out that arbitrary randomization is not safe because it is easy to breach the privacy protection it offers. They proposed random matrix based spectral filtering techniques to recover the original data from the perturbed data. Their experiments revealed that in many cases, random perturbation techniques preserve very little privacy. Storing users' profiles on their own side and running the recommender system in a distributed manner without relying on any server is another approach proposed in [12], where the authors proposed only transmitting similarity measures over the network and keeping users' profiles secret on their side to preserve privacy. Although this method eliminates the main source of threat against user's privacy, it requires higher cooperation among the users to generate useful recommendations. The work in [13] stated that existing similarities are deemed useless as traditional user profiles are sparse and insufficient. Recommender systems need new ways to calculate user similarities. They utilize interpersonal trustworthiness to describe the relationship between two users. The authors in [14] show the correlation between similarity and trust and how it can elevate movie recommendation accuracy.

3 Threat Model

In this work, we assume that an adversary aims to collect users' preferences in order to identify and track users. Thus, we consider our main adversary to be an untrusted PRS to which users send their preferences. We do not assume the PRS to be completely malicious. This is a realistic assumption because PRS needs to accomplish some business goals and increase its revenues. PRS can construct the profiles of the users based on the requests sent. Hence, the problem we are tackling has two sides; we want to detain the ability of the adversary to identify users based on a set of identifying interests and thus track them by correlating these data with data from other publicly-accessible databases and in the same time we want to prevent the adversary from profiling the users through their network identity and therefore invade their privacy. Intuitively, the system privacy is high if PRS is not able to reconstruct the real users' preferences based on the information available to it. Other adversaries are malicious users trying to collect preferences information about others. Malicious users can eavesdrop and collect preferences while being aggregated. So, while hiding our identity from the recommender service, it should not be revealed to other users sniffing the network.

4 Private Recommender Service for IPTV Network Scenario

We extend the scenario proposed in [15-20], where a private recommender service (PRS) is implemented as an external third party server and users give their preferences to that server in order to receive referrals. *EMCP* preserves users' privacy by utilizing three mechanisms: trust based obfuscation, aggregation and threshold encryption. The basic idea for a recommendation based on *EMCP* is that the user who needs recommendation will form a group with other participants in the IPTV network who decided to join his recommendation process. Then, the group members elect highly reputable peers (that we call super-peer) to aggregate their preferences they are willing to share into profiles. The super-peers will cooperate to achieve privacy by encrypting collected profiles using threshold homomorphic encryption in order to permit particular operations to be performed on encrypted data without need for prior decryption and then submit these aggregates to PRS in order to produce referrals. The encrypted profiles hide the identities of the participants, and thus hamper the ability for the untrusted PRS to profile and track users that invade their privacy. However, participants cannot trust each other as well and hence the aggregation process should not expose their preferences. Hence, we proposed a trust based obfuscation mechanism to obfuscate preferences prior submission to super-peers.

Our solution relies on a two stage concealment process, the first stage is trust based obfuscation and it takes place at participant side to hide extracted preferences prior their submission to super-peers. Then the second stage is threshold homomorphic encryption and it takes place at super-peers to hide collected profiles prior their submission to PRS. The overall process might be described as follows: upon receiving a request from the target user, a group of participants is formed that is managed by an elected super-peer. Super-peers negotiate with both the target user and PRS to express their privacy practices for the data collection and usage via P3P policies which are XML statements that answers questions concerning purpose of collection, the recipients of these profiles, and the retention policy. After receiving P3P policy& request, *EMCP* ensures that the extracted preferences for specific request do not violate the privacy of its host by checking whether there is an APPEL privacy preference corresponding to that given P3P policy, and then it starts collecting preferences that fulfil the request and in the same time satisfies the extracted APPEL preferences. The extracted items' ratings are obfuscated using a trust based obfuscation mechanism provided by *EMCP,* such that each item's rating is obfuscated based on the privacy preferences of its owner and estimated trust level with the target user. Furthermore, items identifiers and meta-data are hashed using locality-sensitive hashing. This step prevents the super-peers from knowing each participant's raw ratings for different items identifiers. The super-peer collects these obfuscated preferences and computes an aggregation on them, which does not expose individual ratings. Next, the collected profiles are encapsulated using threshold encryption and submitted to PRS to predicate ratings for the referred items that did not consumed before and will be offered in the end to the target-user and participants. The collaborative filtering task at PRS will be reduced to computing addition on aggregated ratings without exposing the raw ratings. Therefore, our solution ensures privacy in the relation between the participants and PRS and in between the participants themselves. In the following section we will describe some enhancements attained using *EMCP*:

1. ***Usage of Pseudonymous for the Profiles:*** The real user's identity is not always required to provide referrals. Users can be identified by anonymous pseudonyms or nicknames, so that the binding of nickname and the real life identity is not always manifested.
2. ***User Private Data Store at the Client:*** Shifting from the approach of storing the user profiles in the server side to the one of storing the profiles on the clients' STBs helps reducing the privacy concerns. One key aspect is keeping the profiles encrypted to avoid people having access to the client's machine or malware that looks for user profiles.
3. ***Request-Oriented Collection:*** Upon receiving a request from the target user, query rewriter and preference checker assures that learning agent extracts only the required preferences from user's profile for a particular request the user is engaged in. The key point relies on knowing what kind of data is required for a given request that can contribute to improve the performance of the recommendation, because the recommender service does not provide recommendation based on one user's full profile information (e.g.: other users' preferences might not be relevant to the request). Likewise, once a user completes a particular request, he/she may no longer be interested in receiving recommendations related to that request for a period.
4. ***Communication through Anonymous Networks:*** internet records containing IPs, etc stored at service providers, contain information that permit the identification of user when submitting their obfuscated preferences to the node that requested recommendation. *EMCP* employ anonymous communication to hide the network identity for the participants by routing the submission of their obfuscated preferences through relaying nodes in an anonymous communication network before sending them to Super-peers. The main challenge for *EMCP* is to tune up and optimize the performance of the anonymous network while maintaining the user anonymity, we employed the path selection algorithm presented in [18] to enhances the anonymous network performance. Figure (1) shows the architecture of our approach.

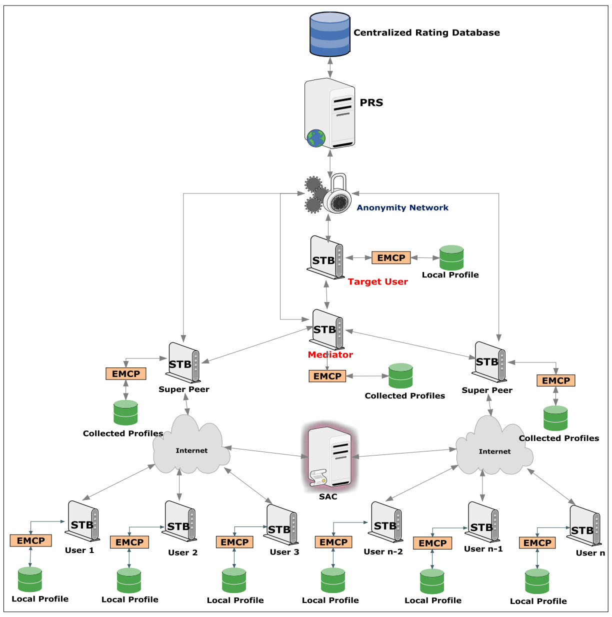

Figure 1: *EMCP* Middleware with Third Party Private Recommender Service.

Our solution relies on the hierarchical topology proposed in [21]; per each request participants are organized into peer-groups managed by super-peers. Electing super-peers is based on negotiation between the participants and security authority centre .The security authority centre (SAC) is a trusted third party responsible for making an assessment on those super-peers according to the participants' reports and periodically updating the reputation of each super-peer based upon it. Reputation mechanisms are employed to elect suitable super-peers based on estimating values for user-satisfaction, trust level, processing capabilities and available bandwidth, further details and information on complex reputation mechanisms can be found in [22]. When a problem occurs with a specific super-peer during the recommendation process, a participant can report it to SAC. After investigation, the assessment of the super-peer will be degraded. This will limit the chance for electing it as a super-peer in the future. On the other hand, successful recommendation processes will help upgrade the super-peer reputation. An IPTV provider can offer certain benefits (like free content, prizes,... etc) for those participants who have a sustained success rate as a super-peer.

Our solution depends upon the set top box (STB) device at the user side. STB is an electronic appliance that connects to both the network and the home television. With the advancement of data storage technology each STB is equipped with mass storage, e.g. Cisco STB. *EMCP* components are hosted on STB; Moreover STB storage stores the user profile. On the other hand, PRS maintains a centralized rating database that is used to provide referrals if the number of participants in group fall below a certain threshold. PRS is the third-party entity recruited by the IPTV network provider to operate referrals by consolidating the information received from multiple sources.

5 Proposed Solution

In the beginning, we want to introduce the notions of privacy and trust within our framework, we need to confirm what we mean by privacy and trust first. To define privacy and trust in our terms, we first approach the notion of privacy in following terms: “A participant who wants to join recommendation request in a network of users, does not has to reveal raw ratings in his/her profile during the recommendation process and elected super-peers does not wish PRS to learn any raw ratings in the collected profiles they provide”. While in the context of this paper, trust is interpreted as “a user’s expectation of another user’s competency in providing ratings to reduce its uncertainty in predicating new items’ ratings [23]”. In our framework, the notion of privacy surrounding the disclosure of users’ preferences and the protection of trust computation between different users are together the backbone of our solution. We apply a trust based obfuscation mechanism at participant side, which produces different copies of items’ ratings based on the various trust levels with target user. The trust computation is done locally over the obfuscated participant’s preferences, and then recommendation is served using secure multi-party computation protocol. Utilizing trust heuristic as input for both group formation and obfuscation process has been of great importance in mitigate some of malicious insider attacks such as infesting the trust computation results. As future work, we plan to investigate miscellaneous insider attacks and strengthen our framework against them.

In the next sub-sections, we will present our proposed middleware for protecting the privacy of users’ preferences. Figure (2) illustrates the *EMCP* components running inside user’s STB. *EMCP* consists of different co-operative agents. A Learning agent captures user interests about miscellaneous items explicitly or implicitly to build a rating database and meta-data database. The local obfuscation agent implements a trust based obfuscation mechanism to achieve user privacy while sharing his/her preferences with super-peers or PRS. The encryption agent is only invoked if the user is acting as a super-peer in the recommendation process; it executes *SR* protocol on the collected profiles. These mechanisms act as wrappers that conceal preferences before they are shared with any external entity. Since the database is dynamic in nature, the local obfuscation agent periodically desensitizes the updated preferences, and then a synchronize agent forwards them to the PRS upon owner permissions. Thus recommendation can be made on the most recent ratings. Moreover, synchronize agent is responsible for calculating & storing parameterized paths in anonymous network that attain high throughput[18], which in turn can be used in submitting preferences anonymously. The policy agent is an entity in *EMCP* that has the ability to encode privacy preferences and privacy policies as XML statements depending on the host role in the recommendation process. Hence, if the host role as a “super-peer”, the policy agent will has the responsibility to encode data collection and data usage practices as P3P policies via XML statements which are answering questions concerning purpose of collection, the recipients of these profiles, and the retention policy. On the other hand, if the host role as a “participant” policy agent acquires the user’s privacy preferences and express them using APPEL as a set of preferences rules which are then decomposed into set of elements that are stored in a database called “privacy preferences” as tables called “privacy meta-data”. These rules contain both a privacy policy and an action to be taken for such privacy policy, in such way this will enable the preference checker to make self-acting decisions on objects that are encountered during data collection process regarding different P3P policies (e.g.: privacy preferences could include: Certain categories of items should be excluded from data before submission, Expiration of purchase history, Usage of items that have been purchased with the business credit card and not with the private one, Generalize certain terms or names in user’s preferences according to defined taxonomy, Using synonyms for certain terms or names in user’s preferences , suppressing certain items from the extracted preferences and insert dummy items that have same feature vector like the suppressed ones as described in [24], limiting the potentially output patterns from extracted preferences etc in order to prevent the disclosure of sensitive preferences in user’s profile). Query Rewriter rewrites the received request constrained by privacy preference for its host.

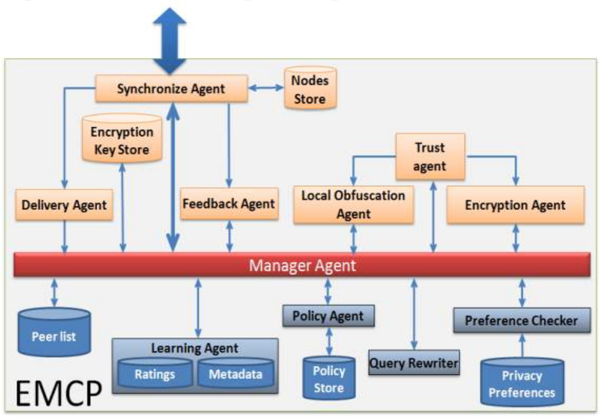

Figure 2: EMCP Components.

Figure (3) shows the participants interactions with super-peers and PRS. The recommendation process in our solution operates as follows:

1. The learning agent collects the user’s interest about different items which represent his profile. The local profile is stored on two databases, the first one is the rating database that contains (item_id, rating) and the second is the meta-data database that contains the feature vector for each item [24] (item_id, feature1, feature2, feature3). The feature vector can include genres, directors, actors and so on. Both implicit and explicit ways for information collection [25] are used to construct these two databases and maintain them.
2. As stated in [16], the target user broadcasts a message to other users in the IPTV network requesting recommendation for a specific genre or category of items. Thereafter, the target user selects a set of his

preferences to be used later in the computation of trust level at the participant side. So as to hide the items identifiers and meta-data from other participants, The local obfuscation agent uses locality-sensitive hashing (LSH) [26] to hash these values. One interesting property for LSH is that similar items will be hashed to the same value with high probability. Super-peers and PRS are still able to perform computation on the hashed values using appropriate distance metrics like hamming distance or dice coefficient. Simultaneously, local obfuscation agent sanitizes items' ratings using trust based obfuscation. Finally, the target user dispatches these obfuscated items' ratings along with their associated hashed values to the Individual users who have decided to participate in the recommendation process.

3. Each group of participants negotiates with SAC to select a peer with the highest reputation as a "super-peer" which will act as a communication gateway between the PRS and the participants in its underlying group.
4. Each super-peer negotiates with both the target user and PRS to express its privacy policies for the data collection and usage process via P3P policies which are XML statements that answers questions concerning purpose of collection, the recipients of these profiles, and the retention policy. Thereafter, super-peers engage in key distribution phase of the *SR* protocol, at the end of this phase each super-peer will possess a share of the decryption key along with the complete encryption key to encrypt the collected profiles. The encrypted profiles can only be decrypted only if any subset consisting of a threshold t of super-peers cooperate.
5. At the participant side, the manager agent receives the request from the target user along with the P3P policy form the elected super-peer; then it forwards P3P policy to preference checker and the request to query rewriter. The preference checker ensures that the extracted preferences for a specific request do not violate the privacy of its host by checking whether there is an APPEL preference corresponding to the given P3P policy and sends it to the query rewriter. The user's preferences can be transferred or collected only if the purpose of statement for the collectors satisfies the privacy preferences. The query rewriter will have knowledge about privacy preferences related to current request via APPEL preference then it rewrites the received request constrained by the privacy preference for its host in order to only retrieve the preferences that the host agrees to share as well as prevent the disclosure of confidential preferences in the participant's profile. This step enable the participant to decide when the recommendation takes place, which information should be collected and for which purpose. This step will ensure the privacy principles compliance and put the user in control the information that is part of their profiles. The modified request is directed to the learning agent to start collecting preferences that could satisfy the modified query. The manager agent ensures that the collected preferences compliance with the collection data principle, as only the required preferences for the particular request the user is engaged in, is extracted for the local obfuscation process.
6. In the meanwhile, the trust agent calculates approximated interpersonal trust between its host and the target user based on the received preference. It is done in a decentralized fashion using the entropy definition proposed in [23] at each participant side. The entropy value becomes lower as the users' ratings are more consistent, which is similar to the definition of trust previously stated. $\forall_{j=1}^{n} T\left(u_a, u_{b_j}\right)$ is the estimated trust between the target user u_a and participant u_{b_j}. the whole process can be described using the following steps:
   i. Each participant $\forall_{j=1}^{n} u_{b_j}$ determines a subset of his/her items' ratings that will be required for recommendation process. Then the participant utilizes shared items rated by both of u_a, u_{b_j} for the trust computation. Determining shared rated items is done by matching the received items' hash values from target user u_a with his/her local items' hash values.
   ii. Participant u_{b_j}computes the trust level using equation $T\left(u_a, u_{b_j}\right) =$
   $$\frac{Entropy(u_a) - Entropy\left(u_a \middle| u_{b_j}\right)}{Entropy(u_a)} \qquad (1)$$
   $$= \frac{\left(1 - \frac{\log N}{\log ZN}\right) + \frac{1}{N \log ZN}\left(\sum_{i=1}^{Z}\sum_{j=1}^{Z} n_{ij} \log n_{ij} - \sum_{i=1}^{Z} n_i \log n_i\right)}{1 - \frac{1}{N \log ZN}\sum_{i=1}^{Z} n_i \log n_i}$$
   Equation (1) is an adapted formalization of trust as proposed in [23] where *Z* denotes the number of states of rated values and *N* is the total number of rating times. For example if *Z=6* and *N=20* when 20 ratings are made with 1 to 6 integer valued scores. Employing entropy to select trustworthy neighbours achieves an improvement in the group formation and rating predication. The enhancement in rating predication is stemmed from trust propagation, so if $u_{b_{j=x}}$ is selected as a trustworthy user and he/she does not have a rating for the item to be predicted, a trustworthy user $u_{b_{j=y}}$ of user $u_{b_{j=x}}$ can also be used for the predication.
   iii. Each participant $\forall_{j=1}^{n} u_{b_j}$ sends his/her calculated trust value to the super-peer. The Estimated trust values are forwarded to both the super-peers and PRS.
   iv. Each participant $\forall_{j=1}^{n} u_{b_j}$ sends this trust value to the local obfuscation agent to adjust the obfuscation level with trust level, in other words, we correlate the obfuscation level with different levels of trust, so the more trusted a target user is, the less obfuscated copy of users' preference he can access. The local obfuscation agent executes enhanced value-substitution (*EVS*) algorithm on items' ratings that are required in the recommendation process. Moreover the local

obfuscation agent hashes their identifiers and meta-data using LSH. The level of obfuscation is determined using the trust level with the target user, and then participants submit their obfuscated preferences to the super-peers of their group. Anonymous communication [18] utilized to hide the network identities of group members when submitting their obfuscated preferences to the super-peers.

v. Finally, the policy agent audits the original and modified requests plus estimated trust level and P3P policy with previous requests; this step allows *EMCP* to prevent multiple requests that might extract sensitive preferences. In such a case, if the target user requests same data twice, its trust level will be reduced, in which will increase the level of the obfuscation in the extracted preferences. This step will cause extracted preferences appear as a completely different set of preferences to the target user.

7. Upon receiving the obfuscated preferences from the participants, each super-peer filters the received preferences based on the trust level of their owners such that $T\left(u_a, u_{b_j}\right) > \theta$ where θ is a minimum trust threshold value defined by the target user or PRS. Then, each super-peer collects the participants' pseudonyms and builds a group rating profile such that all the <hashed value, rating> elements belonging to similar items are grouped together. This allows the computing of the items popularity curve at each super-peer. The super-peer can seamlessly interact with the PRS by posing as an end-user and has a group profile as his own profile. Each super-peer $\forall_{x=1}^{k} SP_x$ calculates the following intermediate values for each user in the *N*-neighbourhood of target user $\forall_{j=1}^{n} u_{b_j} \in \text{Neighbor}(u_a)$,

   Then $\forall\ q = 1 \dots T \quad \widetilde{r_{u_{b_j},q}} = r_{u_{b_j},q} - \overline{r_q}$

$$\widehat{r_{q,u_{b_j}}}^{x} = \frac{T\left(u_a,u_{b_j}\right) * \widetilde{r_{u_{b_j},q}}}{T\left(u_a,u_{b_j}\right)} \qquad (2)$$

   Where $r_{u_{b_j},q}$ is the rating value of participant u_{b_j} for item q. $\overline{r_q}$ is the average rating for item q in each items' cluster. Next, each super-peer encrypts theses intermediate ratings $\widehat{r_{q,u_{b_j}}}^{x}$ using the encryption key pk. Finally, the super-peer submits these ratings along with their associated hashed values to PRS, which in turn collects them to produce final referrals.

8. Upon receiving the encrypted ratings $\forall_{x=1}^{k}\forall_{j=1}^{n}\varepsilon_{pk}\left(\widehat{r_{q,u_{b_j}}}^{x}\right)$ from all super-peers, PRS stores them along with their participants' pseudonyms and hashed values in the centralized rating database. The rating predication phase is performed using the additive homomorphic property of the threshold paillier encryption as the required computations are additive. Thus, PRS executes an additive operation on the encrypted rating profiles without decrypting them so the private data of multiple super-peers can be preserved during the computation. Calculating the predicted rating for referrals done as shown in equation (3):

$$p_{u_a,q} = \varepsilon_{pk}\left(\overline{r_{u_a}}\right) * \left(\prod_{j=1}^{n} \varepsilon_{pk}\left(\widehat{r_{q,u_{b_j}}}^{x}\right)\right)$$
$$= \varepsilon_{pk}\left(\overline{r_{u_a}} + \left(\sum_{j=1}^{n} \widehat{r_{q,u_{b_j}}}^{x}\right)\right) \qquad (3)$$

   Notice that the result will be equal to the weighted sum of the participants' rating plus the average rating of the target user r_{u_a}.Super-peers uses the reblinding property of the paillier encryption to prevent PRS and target user from obtaining any knowledge of $\widehat{r_{q,u_{b_j}}}^{x}$ values by trying a few possible values.

9. PRS forwards the encrypted referrals list along with their predicated ratings to super-peers which in turn perform threshold decryption on these results. The threshold decryption process requires that at least t of the super-peers are honest. Only when the required number of super-peers cooperates, they can perform decryption using their local share of the private key, and then they will be able to have the final referrals list for the entire group. Super-peers publish the final list to the target user and participants. Finally, each participant report scores about the elected super-peer of his group and target-user to SAC, which helps to determine reputation of each entity involved in referrals generation.

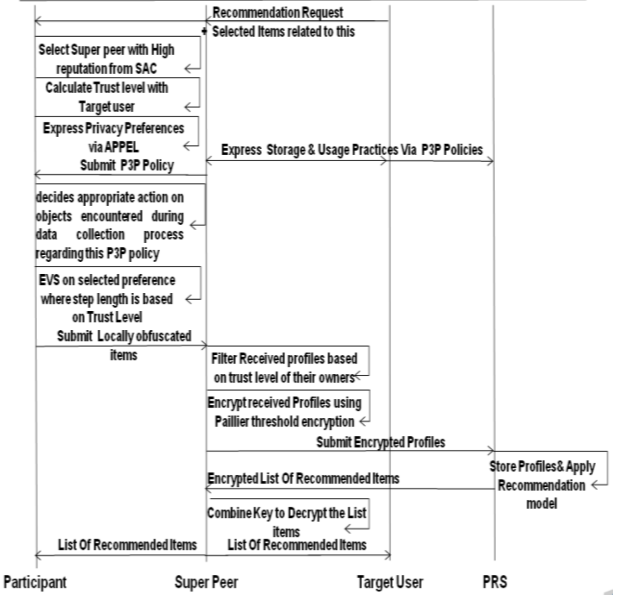
Recommendation Request
Selected Items related to this
Select Super peer with High reputation from SAC
Calculate Trust level with Target user
Express Privacy Preferences via APPEL
Submit P3P Policy
Express Storage & Usage Practices Via P3P Policies
decides appropriate action on objects encountered during data collection process regarding this P3P policy
EVS on selected preference where step length is based on Trust Level
Submit Locally obfuscated items
Filter Received profiles based on trust level of their owners
Encrypt received Profiles using Paillier threshold encryption
Submit Encrypted Profiles
Store Profiles& Apply Recommendation model
Encrypted List Of Recommended Items
Combine Key to Decrypt the List items
List Of Recommended Items
List Of Recommended Items
Participant
Super Peer
Target User
PRS

Figure 3: Interaction Sequence Diagram.

6 Proposed Two Stage Concealment Process

In the next subsections, we present our two stage concealment process used in *EMCP* to disguise the user items' ratings in way that secure the user's preferences in the untrusted PRS with minimum loss of accuracy. In our framework, each user has two datasets representing his/her profile. A local profile: represents the actual ratings of the user for different items; it is stored on his STB. Each user disguises this local profile before sending it to super-peer. An encrypted centralized profile: this is the output of the two-stage concealment process that stored at PRS, the user gets recommendation directly from the PRS based on the previously collected profiles. We perform experiments on real datasets to illustrate the applicability of our mechanisms and the privacy and accuracy levels achieved by using them.

6.1 Cryptography Tools

Using additively homomorphic cryptosystem permit the computation of linear combinations of encrypted data without need for prior decryption, such that PRS can combine received encrypted rating profiles into a new ciphertext that is the encryption of the sum of the ratings of the original ratings. Formally, an encryption schema $\varepsilon_{pk}(.)$ denotes the encryption function with encryption key pk and $D_{sk}(.)$ denotes the decryption function with decryption key sk. Additive homomorphic cryptosystem possesses the following properties:

1. Given the encryption of plaintexts m_1 and m_2, $\varepsilon_{pk}(m_1)$ and $\varepsilon_{pk}(m_2)$. The sum $m_1 + m_2$ can be directly computed as $\varepsilon_{pk}(m_1 + m_2) = \varepsilon_{pk}(m_1) * \varepsilon_{pk}(m_2)$.
2. Given a constant k and the encryption of m_1, $\varepsilon_{pk}(m_1)$. The multiplication of k with the plaintext m_1 can be directly computed as $\varepsilon_{pk}(k.m_1)$= $\varepsilon_{pk}(m_1)^k$.

Paillier [27] proposed a probabilistic asymmetric algorithm for public key cryptography that is an example of an efficient additively homomorphic cryptosystem, this scheme is further extended by [28] with a threshold versions, but required the use of a trusted dealer to distribute the keys to the participants. The reliance on a trusted dealer was lifted in [29] to ensure that no single party or coalition of less than specific participants can recover the encrypted values. In designing *SR* protocol, we require a fully distributed key generation protocol. In particular, the coalition between PRS or target user with any super-peer within the group should not be able to decrypt the whole collected profiles submitted to PRS, but it only reveals the obfuscated profiles collected by this super-peer. Therefore neither can be used as a trusted "dealer" for key generation. Thus, we employ a fully distributed threshold cryptosystem, Since it is desirable to distribute trust between numerous super-peers and no single super-peer is assumed to be fully trusted, then the decryption key sk is shared among a number P of super-peers, and encrypted profiles can only be decrypted only if any subset consisting of a threshold t of super-peers cooperate but no subset smaller than t can perform decryption. Moreover, with the additively homomorphic property of Paillier schema it permits *SR* protocol to perform secure aggregation and predication over encrypted rating profiles. We assume a semi-honest model for the super-peers. Hence, we do not require zero-knowledge proofs (ZKPs) for the various cryptographic operations from the participants. We will briefly present the distributed paillier threshold cryptosystem below.

Key Generation

In this step, each super-peer $\forall_{i=1}^{n} SP_i$ generates n additive shares of two $\kappa/2$-bit strong primes, such that each super-peer have share p_i and q_i. Then use the method proposed in [29] to compute $N = pq$, $\lambda = lCM\ (p-1, q-1)$, $g = N + 1$ such that $p = \sum_{i=1}^{n} p_i$, $\mathrm{q} = \sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{q_i}$, also d such that $\mathrm{d} \equiv 1 \bmod \mathrm{N}$ and $\mathrm{d} \equiv 0 \bmod \lambda$. The public key $pk = (N, g)$ and the private key $sk = d.$ Note that, super-peers perform biprimality test in [30] for checking if N is a product of two primes in a distributed way. If the test fails, the protocol is restarted

Key Sharing

The private key sk is shared among n super-peers with the Shamir scheme as $\mathrm{t} - 1$ degree polynomial where each party obtain (t, n) share of d : Let $\mathrm{a_0} = \mathrm{d}$, and randomly choose $\mathrm{a_i}$ in $\{0, \ldots \mathrm{N} - 1\}$ and set $\mathrm{f(X)} = \sum_{\mathrm{i}=0}^{\mathrm{t}} \mathrm{a_i X^i}$. The share $\mathrm{s_i}$ of the ith super-peer $\mathrm{SP_i}$ is f(i) mod N.

Encryption

To encrypt a message $\mathrm{M} \in \mathrm{Z_N}$ with public key , randomly choose $\mathrm{r} \in \mathrm{Z_N^*}$ and compute $\mathrm{C} = \mathrm{g^M r^n} \bmod \mathrm{N}^2$.

Share Decryption

To decrypt C, each super-peer $\mathrm{SP_i}$ computes the decryption share $c_i = c^{2\Delta si} mod\ N^2$, where $\Delta = t!$ using his/her secret share $\mathrm{s_i}$. And finally, if $\mathrm{t} + 1$ valid shares are available, they can be combined to recover M as described in End decryption.

End Decryption

Let S be a set of $\mathrm{t} + 1$ valid shares. Compute

$$M = L\left(\prod_{i \in S} c_i^{2\lambda_i} mod\ N^2\right) \frac{1}{4\Delta^2} mod\ N$$

Where $\lambda_i = \Delta \prod_{i' \epsilon S \setminus i} \frac{-i}{i-i'}$, See [29] for more details on the correctness of the scheme and for proofs of security.

6.2 Local Obfuscation using Enhanced Value-Substitution (*EVS*) Algorithm

We propose a novel algorithm for obfuscating the users' ratings before sending them to the super-peers. This algorithm is called *EVS*, which has been designed especially for the sparse data problem we have here. Moreover the algorithm tunes its obfuscation parameters based on trust level. The available anonymisation algorithms perform single obfuscation levels for all participants and release one obfuscated copy for all of them which result in increasing data distortion and construction of inaccurate recommendation model. The key idea for *EVS* is based on the work in [31] that uses Hilbert curve as a dimensionality reduction tool to create a cloaking regions to attain privacy for users. Hilbert curve also has the ability to maintain the association between different dimensions. In this subsection, we extend this idea as following, we also use Hilbert curve to map m-dimensional profile to 1-dimensional profile then *EVS* discovers the distribution of that1-dimensional profile. Finally, we perform perturbation based on that distribution in such a way to preserve the profile range that led to providing accurate results when performing rating predication. The output of our obfuscation algorithm should satisfy two requirements:

- Reconstructing the original profile from the obfuscated profile should be difficult, in order to preserve privacy.
- Preserving the distances of the data to achieve accurate results for the recommendation.

The steps for *EVS* algorithm consists of the following:

1. We denote the collected m-dimensional user preferences as dataset D of c rows, where each row is a sequence of m dimensions $A = A_1, A_2, A_3, A_4 \dots\dots, A_m$.
2. Trust level values are divided to a number of intervals defined by the user, associated with each interval an order k value. *EVS* chooses a value for order k according to the trust level associated with the target user.
3. *EVS* divides the m-dimensional dataset D into grids of order k as shown in [31, 32]. For order k, the range for each dimension divided into 2^k intervals.
4. For each dimension $\forall_{i=1}^{m} A_i$ of the collected preferences D:
   - Compute the k-order Hilbert value for each data point $\forall_{x=1}^{c} a_{ix}$. This value represents the index of the corresponding interval where it falls in.
   - *EVS* sort the Hilbert values from smallest to biggest, then use the step length (a user defined parameter) to measure whether any two values are near from each other or not. If these values are near, they are placed in the same partition $\forall_{v=1}^{k} k_{iv}$.

   These two steps iterates for all m-dimensions. The final result from these steps is k partitions for each dimension denoted as $\forall_{i=1}^{m} \forall_{v=1}^{k} C_{iv}$
5. *EVS* constructs a N shared nearest neighbour sets S_r where $r = 1 \dots . N$ as in [33] from different partitions with a new modified similarity function as following, two partitions in different dimensions C_{iv}, C_{i+1v} form a shared nearest neighbour set S_r if they share k-number of common elements such that $S_r = C_{iv} \cup C_{i+1v}$
6. For each newly created set S_r, *EVS* calculates its interquartile range. Then, for each point $a_i \in S_r$ generate a uniform distributed random point n in that range that can substitutes a_i.
7. Finally, the new set $D' = \cup_{r=1}^{N} S_r$ is sent to Super-peer.

6.3 Secure Recommendation Protocol (*SR*)

We proposed a protocol that enables PRS to calculate predicted ratings from the encrypted rating profiles. We called this protocol secure recommendation protocol (*SR*). *SR* protocol starts with the selection of super-peers using SAC as it is heavily relies on the underlying network topology; also it requires a set of super-peers to aggregate all participants' preferences at the bottom of the hierarchy into profiles in order to remove any possibility of a single super-peer being the bottleneck. To achieve reasonable efficiency, super-peers reserve the ability to independently reweight items' ratings based on trust values and omit the ones with low trust values, where such centralized computation can make the most significant difference. Moreover, they compute aggregated items' ratings from the obfuscated ratings received from their participants. Thereafter, super-peers engage in distributed key generation process using distributed threshold cryptosystem to generate public key to encrypt these profiles before submitting them to PRS. This key generation process will leave each super-peer with a share of the private key along with the complete public key. This makes sure that no single super-peer to able decrypt the profiles taken from different super-peers or the final referrals list retrieved from PRS. After all the super-peers collect preferences from participants and compute the aggregated ratings profiles, they engage independently in encrypting these results. Then, each super-peer will forward the ciphertext corresponding to the ratings profile over the entire group to the PRS. The PRS starts the rating predication phase on the ciphertext then forward back the results to the super-peers. The super-peers will then perform threshold decryption of these results. Only when the required number of super-peers cooperates, they can perform decryption using their local share of the private key, and then they will be able to have the final referrals list for the entire group. Note that we have focused on the decryption process to make sure that no single super-peer can get the profiles over a subset of super-peers in the group and malicious super-peers in the network are unable to compromise the security of the protocol. Moreover, utilizing fully distributed threshold cryptosystem ensures that all collected profiles become useless after the termination of recommendation process even if an attacker obtains the collected profiles. *EMCP* automatically destroys key shares directly after decrypting the received referrals list, without any explicit action by the participants or any party storing or archiving that data

```
Protocol _SecureRecommendation
Do forever
/* Applied in cases where super-peers are not already defined,
Electing super-peers is based on negotiation between
participants and SAC to select peers with the highest
reputations*/
SuperPeer = selectSP ();
/* Find out who are other super-peer from SAC */
SPList = find SuperPeer ();
/* Check if I am super-peers to start collecting participants'
preferences & generate keys for encryption agent */
If (me == SuperPeer)
    /* Delivery agent listens to receiver channel to collect
    obfuscated preferences from participants associated with
    this super-peer */
    ListenToReceiverChannel (CollectChannel,
    ReceivedObfuscatedPreferences $r_{u_{b_j},q}$ , $T\left(u_a, u_{b_j}\right)$ );
    /* Delivery agent combine the obfuscated preferences on
    the receiver channel if trust level for its participant higher
    than specific threshold value $\theta$ set by the target user */
```

```
If $\mathrm{T}\left(\mathrm{u_a}, \mathrm{u_{b_j}}\right) > \theta$ then store $\mathrm{r_{u_{b_j},q}}$
    /* Delivery agent combine the obfuscated preferences
    on the receiver channel, if there is a change in the local
    preferences or if there is a new preferences received */
    if (LocalObfuscatedPreferences == true ||
    NewReceivedObfuscatedPreferences == true)
    /* Calculates the normalized rating for item q from
    rating of each participant $u_b$*/
    $\forall_{j=1}^{n} u_{b_j} \in Neighbor(u_a)$ calculate $\widetilde{r_{u_{b_j},q}}$
                        $= r_{u_{b_j},q} - \overline{r_q}$
    /* Combine the Received ratings with previously
    collected ratings for each item q*/
    $\forall\ q = 1 \dots T$
      CombinedPreferences $\widehat{r_{q,u_{b_j}}}$ = CombinePreferences
      $\left( \text{LocalObfuscatedPreferences} \left( \frac{T\left(u_a, u_{b_j}\right) * \left(\widetilde{r_{u_{b_j},q}}\right)}{T\left(u_a, u_{b_j}\right)} \right), \text{ReceivedObfuscatedPreferences } \widetilde{r_{u_{b_j},q}} \right)$
    End if
End if
/* Generate public/private Key pair using a distributed
protocol employing all other super-peers. The function
SPDKG () leaves every super-peer with the entire public key
and a share of the private key */
(PublicKey, PrivateKey) = SPDKG(SPList);
/* Initiate the encryption agent to encrypt my combined
preferences with the public key and submit it to PRS */
Submit $\widehat{r_{q,u_{b_j}}}'$ (Enc(PublicKey, CombinedPreferences $\widehat{r_{q,u_{b_j}}}$))
To PRS;
/* PRS receive collected preferences for different super-peers
*/
PRS receives $\mathrm{r}_{\widehat{1,\mathrm{u}_{b_1}{}^1}}{}'$, $\mathrm{r}_{\widehat{1,\mathrm{u}_{b_1}{}^2}}{}'$, …, $\mathrm{r}_{\widehat{2\mathrm{u}_{b_1}{}^1}}{}'$, $\mathrm{r}_{\widehat{2\mathrm{u}_{b_1}{}^2}}{}'$, … , $\mathrm{r}_{\widehat{\mathrm{Tu}_{b_j}{}^k}}{}'$
such that $\mathrm{r}_{\widehat{\mathrm{qu}_{b_j}{}^k}}{}'$ is the encrypted rating for item $\mathrm{q} \in \{1, \dots \dots \mathrm{T}\}$
by user $\mathrm{u_{b_j}}$ $\left( \forall_{j=1}^{n} \mathrm{u_{b_j}}\ |\mathrm{T}\left(\mathrm{u_a}, \mathrm{u_{b_j}}\right) > \theta \right)$ from super-peer
$\mathrm{SP_x}\left( \forall_{x=1}^{k}\ \mathrm{SP_x} \right)$
        /* PRS Calculates Predicated ratings for each unrated
        Item in the collected profiles*/
        For each item $\mathrm{q} = 1$ to T do
        PRS Calculates $\forall_{x=1}^{k} p_{u_a,q} = \left(\mathrm{Enc}(\mathrm{PublicKey}, \overline{r_{u_a}}\right) *$
        $\left(\prod_{j=1}^{n} \mathrm{Enc}(\mathrm{PublicKey}, \mathrm{CombinedPreferences}\ \widehat{r_{q,u_{b_j}}{}^{x}})\right)$
        $\forall_{j=1}^{n} \mathrm{u_{b_j}}$
        End for
/* Upon receiving the list of predicated ratings for referred
items, Target user request super-peers to start decrypt the
entire list */
if (me == SuperPeer) Reclist =
thresholdDecrypt(encryptedratingslist $\left(p_{u_a,q}\right)$, SPList)
End Protocol_SecureRecommendation
--------------------------------------------------------------------
Algorithm selectSP
/* Each participant contact SAC to obtain list of peers of
highest reputation to be elected as super-peer for the group */
Requst(HR_Peerlist);
/* Each super-peer broadcast to the neighbors indicating its
existence as their neighbor*/
broadcast(SP_id);
    /* if participant receives more than super peer id it
    compare P3P policies for each adjacent super-peer &
    select the one with suitable P3P policy to his privacy
    preferences */
    listenToReceiverChannel(defaultChannel, SP_id);
    if (ReceiverPeerId(SP_id) $\neq$ 1)
    Compare(SP_CollectionPolicies);
    PeerGroupJoinRequest(SP_id);
    End if
/*Each super-peer Listens to the receiver channel to form a
group */
listenToReceiverChannel(defaultChannel, numNeighbors);
broadcast(numNeighbors);
listenToReceiverChannel(defaultChannel,PeerGroupCountPair[
]);
superpeer= PeerGroupCountPair[maxIndex].getNeighborID();
return selectSP;
```

7 Proof of Security and Correctness

The proof of security for both *EVS* algorithm and *SR* protocol depends on how much information is leaked during the execution of the prediction phase. At the same time, our proposed mechanisms should output accurate results.

7.1 Privacy Breach Evaluation for *EVS* Algorithm

Privacy breach can be described in terms of how well the original user's ratings can be estimated from the submitted obfuscated ratings. Unlike other techniques, our method generates new data points, whose interpoint distances approximate the original distances. Consequently, points which lie close to one another in the original space mostly remain close to each other in the transformed space. Therefore, it seems theoretically to be more resilient to some potential attacks [34] that exploit the properties of the released data. These attacks are based on how much information about original data is available to the attacker that is obtained through either known input-output and known sample. In the known input-output, attacker knows collection of linearly dependant original data points and points they map in perturbed data. While in known sample, assumes that original data arose as independent samples of multidimensional random vector with unknown probability density function, and the attacker has access to a collection of these independent samples. In *EVS* algorithm, the linear ordering based on Hilbert curve retains the proximity and neighboring aspects of the original data. We define $\mathrm{H_d^N}$ for $\mathrm{N} \geq 1$ and $\mathrm{d} \geq 2$ as the $\mathrm{N^{th}}$order Hilbert curve (defined values based on trust level) for a d- dimensional space. $\mathrm{H_d^N}: [0, 2^{\mathrm{Nd}} - 1] \to [0, 2^{\mathrm{N}} - 1]^{\mathrm{d}}$ as follows : Hilbert value $\mathrm{H} = \epsilon(\mathrm{P})$ for $\mathrm{H} \in \left[0, 2^{\mathrm{Nd}} - 1\right]$, where P is coordinate of each point in $[0, 2^{\mathrm{N}} - 1]^{\mathrm{d}}$. Thereafter, we cluster nearby Hilbert values based on step length (a user-defined parameter) then *EVS* substitutes each point in the group with uniform distributed random point in the same interquartile range for that cluster. Therefore we can consider ϵ as a one-way function if the curve parameters are unknown. These parameters include (starting point, N, step length) are defined at the participant side and any external entity only know the final perturbed data that participant agree to release. As a result, the statistical information from the perturbed data are inconsistent with that from the original data. Therefore, attacks such as those described before would be in efficient in breaching privacy. In addition to that, clustering Hilbert values and substituting each point with random point introduces uncertainty about exact distance between data points, thus will make any distance based attach ineffective.

7.2 Proof of Security & Correctness for *SR* Algorithm

Theorem 1: additive operation performed by PRS in *SR* protocol is correct and accurate without the need of decryption keys.

Proof: based on the first property of additive homomorphic cryptosystem, we can determine that additive operations for encrypted data are correct as follows: given the encryptions $\varepsilon_{pk1}(m_1) = a$ and $\varepsilon_{pk1}(m_2) = b$ where $\forall m_1, m_2 \in \mathbb{Z}_n$, given encryption key $pk2$

$$\varepsilon_{pk2}\left(\varepsilon_{pk1}(m_1)\right).\ \varepsilon_{pk2}\left(\varepsilon_{pk1}(m_2)\right)\ mod\ N^2 =$$
$$\varepsilon_{pk2}(a).\ \varepsilon_{pk2}(b) = (g^a r_1^N).(g^b r_2^N)\ mod\ N^2 =$$
$$g^{a+b}(r_1 r_2)^N\ mod\ N^2 =$$
$$\varepsilon_{pk2}\left(\varepsilon_{pk1}(m_1 + m_2)\right)\ mod\ N^2$$

Based on that, the PRS does not require any decryption key in order to aggregate all encrypted data.

Theorem 2: *SR* protocol computes predicated ratings for each referred item based on similar users' ratings without revealing extra information to any party.

Proof: Since each participant obfuscates his items' ratings and hashes their meta-data before submitting them to the super-peers. Moreover, each super-peer encrypts the collected profiles with the common encryption key and computation is performed on encrypted data and the decryption key is distributed between different super-peers. This makes sure that no single party will be able to decrypt these encrypted profiles taken from different super-peers or the final referrals list retrieved from PRS. This particular property is possible because of the threshold nature of the employed cryptosystem. In our two stage concealment process, the super-peer aggregates all obfuscated preferences then performs intermediate-computations on the obfuscated ratings for each item without having to know their real ratings or identifiers. No party can see extra information during the execution of the *SR* protocol. As for participants, they participate in the recommendation process without knowing other participants' identity. Since not all the participants have the same super-peer nor do they have direct communication with each other. The local profile is secured and can only be viewed by its owner before applying the trust based obfuscation mechanism. In addition, employing reputation techniques to select super-peers with a high success rate in previous recommendation processes ensures the selection of reliable peers that will perform the required phases. PRS cannot see the received profiles as the decryption key is unknown. Furthermore, the decryption process requires a subset consisting of a threshold t of super-peers to cooperate. After PRS generates the final referrals list, PRS submits it to super-peers which in turn perform threshold decryption process. Then they publish this list to all participants.

Theorem 3: Assuming that all parties follow the protocol, *SR* protocol can correctly compute the predicated rating for each referred item.

Proof: When each super-peer encrypts collected rating profiles with encryption key $\varepsilon_{pK}\left(\widehat{r_{q,u_{b_j}}{}^x}\right)$. PRS performs the additive operation on encrypted rating profiles based on paillier's homomorphic cryptosystem as follows:

$$p_{u_a,q} = \varepsilon_{pk}(\overline{r_{u_a}}) + \left(\varepsilon_{pk}(\widehat{r_{q,1}}) + \varepsilon_{pk}(\widehat{r_{q,2}}) + \varepsilon_{pk}(\widehat{r_{q,3}}) + \cdots + \varepsilon_{pk}\left(\widehat{r_{q,u_{b_j}}}\right)\right)$$

$$p_{u_a,q} = \varepsilon_{pk}\left(\overline{r_{u_a}} + \left(\sum_{j=1}^{n} \widehat{r_{q,u_{b_j}}{}^x}\right)\right)$$

$$p_{u_a,q} = \varepsilon_{pk}(\overline{r_{u_a}})\left(\prod_{j=1}^{n} \varepsilon_{pk}\left(\widehat{r_{q,u_{b_j}}{}^x}\right)\right)$$

After the threshold decryption by super-peers, we will obtain the final predicated rating as in equation

$$p_{u_a,q} = \overline{r_{u_a}} * \left(\prod_{j=1}^{n} \widehat{r_{q,u_{b_j}}{}^x}\right)$$

So the result from *SR* protocol is correct.

8 Experiments

In this section, we describe the implementation of our proposed solution. The experiments are run on 2 Intel® machines connected on local network, the lead peer is Intel® Core i7 2.2 GHz with 8 GB Ram and the other is Intel® Core 2 Duo™ 2.4 GHz with 2 GB Ram. We used MySQL as data storage for the users' preferences that acquired by learning agent. The proposed two stage concealment process is implemented in C++ using the MPICH implementation of the MPI communication standard for distributed memory implementation of the *SR* protocol to mimic a distributed reliable network of peers. To implement Paillier encryption scheme, the Number Theory Library (NTL) was used. One practical issue that must be dealt with when using the Paillier cryptosystem is the fact that it cannot naturally encrypt floating-point numbers. Floating-point numbers must be converted to a fixed-point representation. This is done by multiplying them by a large constant C and then truncating the result to an integer. In these experiments, C = 100000. Other methods in [35] can also be used. The experiments presented here were conducted using the Jester dataset provided by Goldberg from UC Berkley [36]. The dataset contains 4.1 million ratings on jokes using a real value between (-10 and +10) of 100 jokes from 73.412 users. The data in our experiments consists of ratings for 36 or more items by 23.500 users. We evaluated the proposed solution from different aspects: privacy achieved, accuracy of results and performance. We used the mean absolute error (MAE) metric proposed in [37]. MAE is one of most famous metrics for recommendation quality. As it measures the predication verity between the predicated ratings and the real ratings, so smaller MAE means better recommendation provided by PRS. To measure the privacy or distortion level achieved using our mechanism, we used the variation of information metric VI [38] to estimate data error. A higher value of VI means a larger distortion between the obfuscated and original dataset, which means a higher level of privacy. The experiments involve dividing the data set into a training set and testing set. The training set is obfuscated then used as a database for PRS. Each rating record in the testing set is divided into rated items t_i and unrated items r_i.The set t is presented to PRS for making predication p_i for the unrated items r_i. For the representation process of the trust calculation, we add the default value 0 for items not rated. In our dataset, the first column of every raw stores how many items are rated by the user, which is necessary for the trust estimation process. We divided trust levels into three intervals [highest, moderate, and lowest] and associated hilbert curve order for each interval. The experiments were performed while keep the number of super-peers $n = 9$, as described earlier they will be responsible for aggregating the data of 23.496 participants. We assume the trust level for all participants to be above the minimum trust threshold θ, which is required for the inclusion in the prediction process. The recommendation process can be initiated by any user that will act as the target-user for the referrals list. The trust level between participants and target-user is calculated locally on their STB devices.

In the first experiment, we want to measure the elapsed time for distributed key generation by varying the encryption key length and number of participants. Therefore we run the function SPDKG in *SR* protocol on 9 super-peers with different key length, and then we measure the elapsed time for distributed key generation and plot results in figure (4). Moreover, we set the key length to 1024 bits with varying number of super-peers (3 to 17) and we plot the elapsed time results in figure (5).

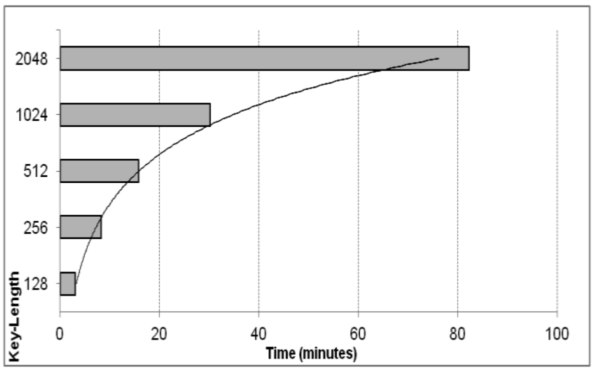

Figure 4: Key generation time for different Key Length.

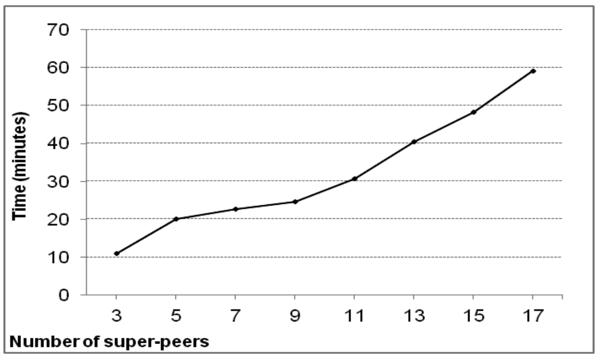

Figure 5: Key generation time for various numbers of super-peers.

In the second experiment, we want to measure the elapsed time for calculating the predicated ratings in *SR* protocol by varying the encryption key length and number of participants. We run the predication phase several times by encrypting all 12.674 records with different key length and distribute them equally on 9 super-peers, then PRS start collecting these records to perform predication phase. The results for elapsed time are shown in figure (6). Moreover, we set the key length to 1024 bits with varying number of super-peers (3 to 17) and we plot the elapsed time results in figure (7).

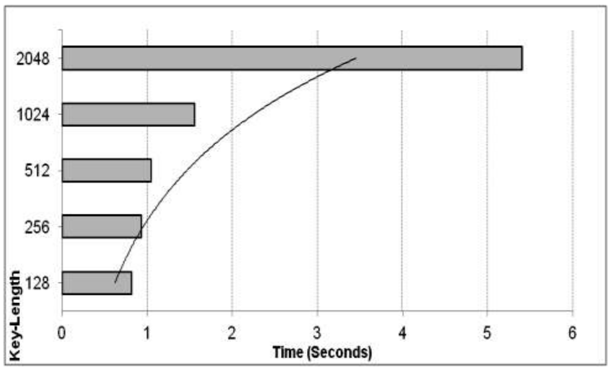

Figure 6: Ratings predication time for different Key Length.

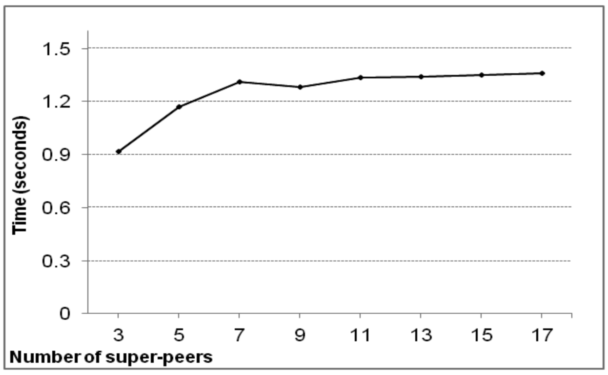

Figure 7: Ratings predication time for various numbers of super-peers.

In the third experiment, we aim to analyze execution time for *SR* protocol for varying set of data sizes. Therefore, we vary the minimum trust threshold to obtain a different number of participants' records in the recommendation process, then we run the *SR* protocol on these aggregated records in sizes of 7.249, 10.572, 12.674, 17.685, and 23.496. As shown in figure (8), the results indicate the elapsed time to perform (encrypt, calculate ratings and decrypt) with 1024 bits key length. The curve scales linearly as it represents the increase of execution time by increasing the data size.

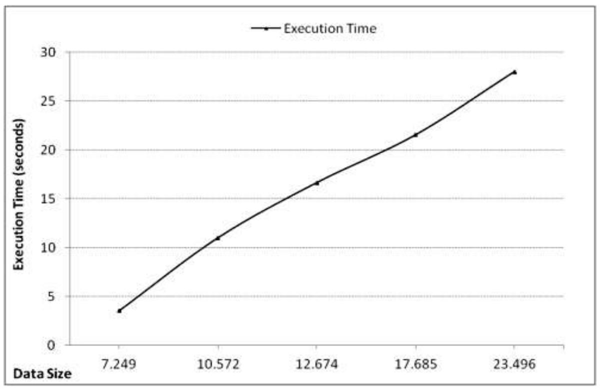

Figure 8: Execution time for different data sizes.

In the first experiment performed on *EVS* algorithm, we measured the relation between different Hilbert curve parameters (order and step length) on the accuracy and privacy levels attained. We mapped the participant's dataset to Hilbert values using orders 3, 6 and 9. We gradually increased the step length from 10 to 80. Figure (9) shows the accuracy of recommendation based on different step length and curve order. We can see that as the order increases, the obfuscated data can offer better predictions for the ratings. Since, with higher values for the curve order, the granularity of the Hilbert curve becomes finer. So, the mapped values can preserve the data distribution of the original dataset. On the other hand, selecting larger step length increases MAE values as large partitions are formed with higher range to generate random values from it, such that these random values substitute real values in the dataset.

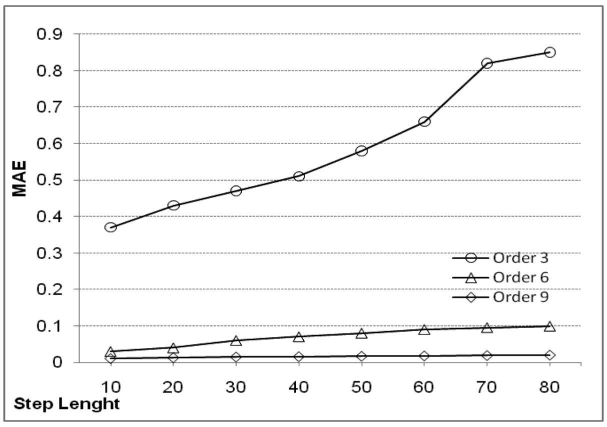

Figure 9: Accuracy level for different step length and orders for *EVS.*

As for the privacy as shown in figure (10), when the order increases a smaller range is calculated within each partition which introduces less substituted values compared with lower orders that attain higher VI values. The reason for this is that larger order divides the m-dimensional profile into more grids, which makes Hilbert curve to better reflect the data distribution. Moreover, we can see that for the same Hilbert curve order the VI values are generally the same for different step length except for order 3, in which VI values has a sharp increase when step length grows from 50 to 60.The effect of increasing step length on VI values is more sensible in lower curve orders as fewer girds are formed and the increase of step length covers more portions of them, which will introduce a higher range to generate random values from it. Based on that, the trust agent employs trust value as an input to tune-up *EVS* parameters in such a way to achieve a trade off between privacy and accuracy.

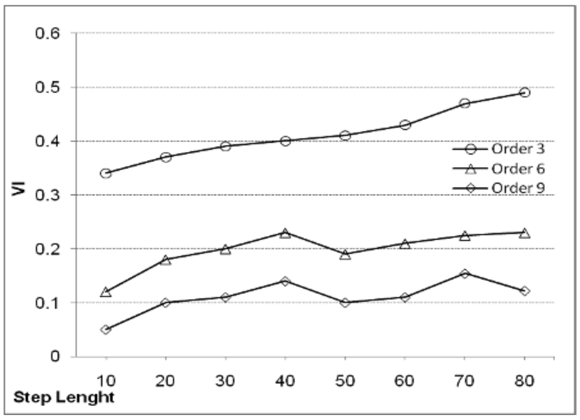

Figure 10: Privacy level for different step length and orders for *EVS.*

We continued our experiments with *EVS* algorithm; we measured the execution time for *EVS* as it is executed locally at the participant's STB box on his profile. The execution time for *EVS* is composed of the time to get partitions based on Hilbert curve and the time to generate random noise. The results for the execution time are shown in figure (11). We can see that as the order of Hilbert curve goes higher, the execution time generally increases than that for a lower order. This growth because of the time consumed in mapping data points to different Hilbert values is dependent on curve order. For different step lengths, the executions time various without substantial trend. As the step length only determines the size of partitions in each dimension; finding these partitions are only dependant on the number of dimensions.

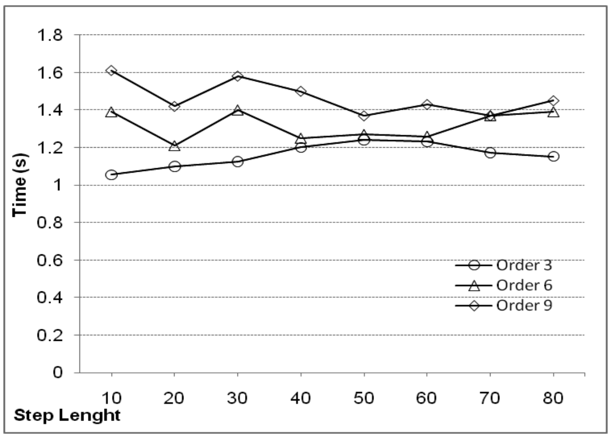

Figure 11: Execution time for different step length in *EVS.*

Finally, we measured the overall recommendation accuracy of our two stage concealment on the same dataset. For *EVS* algorithm, we set the curve order to be 3 (lowest trust level) and the step length to be 10. We first obfuscate different datasets using *EVS* algorithm, then super-peers apply *SR* protocol on these datasets and submit them to PRS. At PRS side, it calculates referrals list then return results back to super-peers which in turn decrypt and publish them. The graph in figure (12) plot MAE values for different data sizes, it clearly shows that the proposed two stage concealment process is very effective in making recommendation and that its privacy preserving nature has marginal impact on the accuracy of recommendation, since *SR* protocol employ homomorphic cryptosystem that preserves the accuracy characteristics of *EVS* algorithm on the dataset. These results indicate two features of the two stage concealment process:

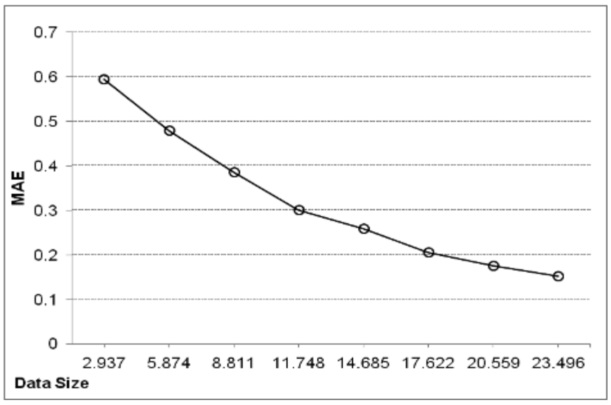

Figure 12: Accuracy of our propsed approch with for different data sizes.

1. Accuracy of the recommendation improves with the increase of collected data, as more diverse ratings produce a reasonable explanation and rank from a reliable sources.
2. Accuracy of the recommendation is reasonable for small datasets, which is highly desirable feature in a dynamic environment like IPTV networks where users' profiles are not large enough.

9 Conclusion and future work

In this paper, we presented our attempt to develop an enhanced middleware for collaborative privacy based on Multi-agent with application to recommender service for IPTV providers. We gave a brief overview of *EMCP* architecture, components and recommendation process. We presented a novel two stage concealment process which provides complete privacy control to participants over their preferences. The concealment process utilizes hierarchical topology, where participants are organized into groups, from which super-peers are elected based on their reputation. Super-peers & PRS use platform for privacy preferences (P3P) policies for specifying their data usage practices. While Participants describe their privacy constraints for the data extracted from their profiles in a dynamically updateable fashion using P3P policies exchange language (APPEL). *EMCP* allows fine grained enforcement of privacy policies by allowing participants to ensure that the extracted preferences for specific request do not violate their privacy by automatically checking whether there is an APPEL preference corresponding to the given P3P policy. Super-peers aggregate the preferences obtained from underlying participants and then encapsulate intermediate values computed on these profiles and then send them to PRS. Trust based obfuscation mechanism is used in the course of participant preferences collection, while the *SR* protocol is used to protect the privacy of collaborative filtering by distributing the participants' preferences between multiple super-peers and encrypting a subset of the aggregated ratings profiles which is useful for the recommendation. We tested the performance of the proposed mechanisms on a real dataset. We evaluated how the overall accuracy of the recommendation depends on data sizes and trust level. The experimental and analysis results show that privacy increases under the proposed middleware without hampering the accuracy of the recommendation. In particular the mean absolute error can be reduced with proper tuning of the trust based obfuscation parameters for a large data sizes. Moreover, utilizing trust levels for obfuscation is an optimization to maintain the utility of the items' ratings. Thus adding the proposed middleware does not severely affect the accuracy of the recommendation based on collaborative filtering techniques.

We realized that there are many challenges in building a privacy enhanced middleware for recommender services. As a result we focused on middleware in a collaborative privacy scenario. A future research agenda will include utilizing game theory to better formulate user groups, sequential preferences release and its impact on privacy of whole profile. We will consider reducing transmission time and the load on the network traffic by adding a secure filtering phase to the *SR* protocol that will allow PRS to exclude items with low predicated rating from the final referrals list. Furthermore it is included to strengthen our middleware against shilling attacks, extending our scheme to be directed towards multi-dimensional trust propagation and distributed collaborative filtering techniques in a P2P environment. Moreover, we need to investigate weighted features vector methods and its impact on released ratings. Such that, the participant not only obfuscates his items' ratings based on the trust level of target-user, but he can also express specific items to be diversely obfuscated with each trust level. We need to perform extensive experiments on other real datasets from the UCI repository and compare our performance with other techniques proposed in the literature. Finally we need to consider different data partitioning techniques as well as identify potential threats and add some protocols to ensure the privacy of the data against those threats.

10 Acknowledgment

This work has received support from the Higher Education Authority in Ireland under the PRTLI Cycle 4 Programme, in the FutureComm Project (Serving

Society: Management of Future Communications Networks and Services).

References

[1] K. Kawazoe, et al., "Platform Application Technology Using the Next Generation Network," NTT2007.
[2] L. F. Cranor, "'I didn't buy it for myself': privacy and Ecommerce personalization," in Designing personalized user experiences in eCommerce, ed: Kluwer Academic Publishers, 2004, pp. 57-73.
[3] M. d. Gemmis, et al., "Preference Learning in Recommender Systems," presented at the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD), Slovenia, 2009.
[4] F. McSherry and I. Mironov, "Differentially private recommender systems: building privacy into the net," presented at the Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, Paris, France, 2009.
[5] A. Esma, "Experimental Demonstration of a Hybrid Privacy-Preserving Recommender System," 2008, pp. 161-170.
[6] J. Canny, "Collaborative filtering with privacy via factor analysis," presented at the Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval, Tampere, Finland, 2002.
[7] J. Canny, "Collaborative Filtering with Privacy," presented at the Proceedings of the 2002 IEEE Symposium on Security and Privacy, 2002.
[8] H. Polat and W. Du, "Privacy-Preserving Collaborative Filtering Using Randomized Perturbation Techniques," presented at the Proceedings of the Third IEEE International Conference on Data Mining, 2003.
[9] H. Polat and W. Du, "SVD-based collaborative filtering with privacy," presented at the Proceedings of the 2005 ACM symposium on Applied computing, Santa Fe, New Mexico, 2005.
[10] Z. Huang, et al., "Deriving private information from randomized data," presented at the Proceedings of the 2005 ACM SIGMOD international conference on Management of data, Baltimore, Maryland, 2005.
[11] H. Kargupta, et al., "On the Privacy Preserving Properties of Random Data Perturbation Techniques," presented at the Proceedings of the Third IEEE International Conference on Data Mining, 2003.
[12] B. N. Miller, et al., "PocketLens: Toward a personal recommender system," ACM Trans. Inf. Syst., vol. 22, pp. 437-476, 2004.
[13] C.-N. Ziegler, et al., "Improving recommendation lists through topic diversification," presented at the Proceedings of the 14th international conference on World Wide Web, Chiba, Japan, 2005.
[14] J. Golbeck and J. Hendler, "FilmTrust: movie recommendations using trust in web-based social networks," in Consumer Communications and Networking Conference, 2006. CCNC 2006. 3rd IEEE, 2006, pp. 282-286.
[15] A. M. Elmisery and D. Botvich, "An Agent Based Middleware for Privacy Aware Recommender Systems in IPTV Networks," in Intelligent Decision Technologies. vol. 10, J. Watada, et al., Eds., ed: Springer Berlin Heidelberg, 2011, pp. 821-832.
[16] A. Elmisery and D. Botvich, "Private Recommendation Service For IPTV System," in 12th IFIP/IEEE International Symposium on Integrated Network Management, Dublin, Ireland, 2011.
[17] A. Elmisery and D. Botvich, "Agent Based Middleware for Maintaining User Privacy in IPTV Recommender Services," in 3rd International ICST Conference on Security and Privacy in Mobile Information and Communication Systems, Aalborg, Denmark, 2011.
[18] A. Elmisery and D. Botvich, "Privacy Aware Obfuscation Middleware for Mobile Jukebox Recommender Services," in The 11th IFIP Conference on e-Business, e-Service, e-Society, Kaunas, Lithuania, 2011.
[19] A. Elmisery and D. Botvich, "Privacy Aware Recommender Service for IPTV Networks," in 5th FTRA/IEEE International Conference on Multimedia and Ubiquitous Engineering, Crete, Greece, 2011.
[20] A. Elmisery and D. Botvich, "Enhanced Middleware for Collaborative Privacy in IPTV Recommender Services " Journal of Convergence, vol. 2, p. 10, 2011.
[21] [21] W. Nejdl, et al., "Super-peer-based routing and clustering strategies for RDF-based peer-to-peer networks," presented at the Proceedings of the 12th international conference on World Wide Web, Budapest, Hungary, 2003.
[22] [22] J. Carbo, et al., "Trust management through fuzzy reputation. Int," Journal in Cooperative Information Systems, vol. 12, p. 135—155, 2002.
[23] [23] H. D. Kim, "Applying Consistency-Based Trust Definition to Collaborative Filtering," KSII TRANSACTIONS ON INTERNET AND INFORMATION SYSTEMS, vol. 3, pp. 366-374, 2009.
[24] [24] A. Elmisery and D. Botvich, "Agent Based Middleware for Private Data Mashup in IPTV Recommender Services," in 16th IEEE International Workshop on Computer Aided Modeling, Analysis and Design of Communication Links and Networks, Kyoto, Japan, 2011.
[25] [25] D. Kelly and J. Teevan, "Implicit feedback for inferring user preference: a bibliography," SIGIR Forum, vol. 37, pp. 18-28, 2003.
[26] [26] P. Indyk and R. Motwani, "Approximate nearest neighbors: towards removing the curse of dimensionality," presented at the Proceedings of the thirtieth annual ACM

symposium on Theory of computing, Dallas, Texas, United States, 1998.

[27] P. Paillier, "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes."

[28] I. Damgård and M. Jurik, "A Generalisation, a Simpli.cation and Some Applications of Paillier's Probabilistic Public-Key System Public Key Cryptography." vol. 1992, K. Kim, Ed., ed: Springer Berlin / Heidelberg, 2001, pp. 119-136.

[29] I. Damgård and M. Koprowski, "Practical Threshold RSA Signatures without a Trusted Dealer Advances in Cryptology — EUROCRYPT 2001." vol. 2045, B. Pfitzmann, Ed., ed: Springer Berlin / Heidelberg, 2001, pp. 152-165.

[30] D. Boneh and M. Franklin, "Efficient generation of shared RSA keys Advances in Cryptology — CRYPTO '97." vol. 1294, B. Kaliski, Ed., ed: Springer Berlin / Heidelberg, 1997, pp. 425-439.

[31] G. Ghinita, et al., "PRIVE: anonymous location-based queries in distributed mobile systems," presented at the Proceedings of the 16th international conference on World Wide Web, Banff, Alberta, Canada, 2007.

[32] A. Reaz and B. Raouf, "A Scalable Peer-to-peer Protocol Enabling Efficient and Flexible Search," ed, 2010.

[33] R. A. Jarvis and E. A. Patrick, "Clustering Using a Similarity Measure Based on Shared Near Neighbors," IEEE Trans. Comput., vol. 22, pp. 1025-1034, 1973.

[34] K. Liu, et al., "An Attacker's View of Distance Preserving Maps for Privacy Preserving Data Mining Knowledge Discovery in Databases: PKDD 2006." vol. 4213, J. Fürnkranz, et al., Eds., ed: Springer Berlin / Heidelberg, 2006, pp. 297-308.

[35] P.-A. Fouque, et al., "CryptoComputing with Rationals Financial Cryptography." vol. 2357, M. Blaze, Ed., ed: SpringerBerlin / Heidelberg, 2003, pp. 136-146.

[36] D. Gupta, et al., "Jester 2.0 (poster abstract): evaluation of an new linear time collaborative filtering algorithm," presented at the Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, Berkeley, California, United States, 1999.

[37] J. L. Herlocker, et al., "Evaluating collaborative filtering recommender systems," ACM Trans. Inf. Syst., vol. 22, pp. 5-53, 2004.

[38] C. Kingsford, "Information Theory Notes," 2009.