Table of Contents

4. Methodology

In this chapter, I propose an efficient privacy-preserving recommender system for users in online social communities. This system model can protect user privacy by ensuring the communication channel between involving users, adversaries will not affect the privacy and security of messages exchanges between them. This system model developed is a group based as it automatically organizes users into groups with diverse interests, so that each user's private interests can be hidden among a set of users against server. A set of pseudo users are created for each user group, each pseudo user delegates a unique interest group, and the union of all pseudo users covers all interests of a user group. The pseudo users communicate with the recommender server on behalf of the real users to get recommendations. The real users can then obtain personalized recommendations based on the server's recommendations to the pseudo users and, without exposing any private data to the server. In this design, four privacy preserving protocols are proposed for different in-group computations, which ensure user privacy.

4.1 Problem formulation:

In this section, we first analyse the user interest privacy issues in online social communities and user-based recommender systems and then propose a high level-design for proposed solution:

In online social communities (e.g. Facebook, Twitter, Google plus), users can post, read or comment on online posts, such as articles, pictures, music or videos. Consider an online social community and its associated recommender server, the following operations are performed by any user in the online community:

- Create a user account
- Post or comment on an item shared or recommended by other users
- Read the content item posted or commented on by other users and
- Finally, request recommendations from recommender server

From the above-mentioned operations, massive and diverse online content is generated by the users. The close interactions between the users have raised new concerns on recommender systems, among which user interest privacy preserving is a key challenge that need to be addressed. Further, public and private information of users in the online social community can be differentiated. Users ‘‘posts’’ and “comments’’ are denoted public, as they are interacted with other users, while users ‘’read’’ information is private as they do not intend to share with other users. Therefore, users ‘’read’’ information should be protected from recommender server and other users during recommendation process.

Given a user u and an online post (*item*) *i*, if *u* has posted/read/commented on *i*, we say u is interested in *i*, then we denote u's rating to *i* as $r_{i,u} = 1$. Otherwise, $r_{i,u} = 0$. Based on the binary ratings (“0” or “1”), the recommender system can generate recommendations based on association rule mining approach and recommend items which meet the minimum support count. In this work, only binary ratings of items from users are considered while other ratings such as 1-5 (e.g. Netflix movie ratings from 1-5) can still be supported. For instance, 1–5 ratings can be normalized by dividing the values by 5, so that “1, 2, 3, 4, 5″ will be normalized to “0.2, 0.4, 0.6, 0.8, 1.0″, respectively. A user rating of an item greater than or equal to 0.6 indicates that user is interested in the rated item.

In frequent itemset mining, recommender systems rely on user interests to mine items that appear frequently and recommend items that achieve minimum support value. The web browsing technologies such as virtual private networks, trusted proxy, help users hide their IP addresses and provide no information to the online service providers. However, these techniques do not help recommender systems to achieve accurate recommendations to the users. So, we need a privacy-preserving recommendation system method which can protect user privacy while still achieving accurate recommendations. To protect individual user's privacy, a high-level system design is proposed. The proposed design, is supposed to provide recommendations to the users without sacrificing the content interest to any party participating in the system. Also, the modelled design targets large scale users in online social communities and is designed to be scalable and efficient. I will summarize key design goals of the model before discussing key components and construction of the model,

- Protect user information during each phase of the design i.e. protect privacy of all involving users in the system in all phases of construction of model.
- Adversaries should not be able to affect the privacy and integrity of information passes through the communication channel.
- The design should be able to converge to a reasonable communication and computation cost.
- Provide accurate recommendations to users.

Design overview:

As illustrated in the figure 1, the proposed design consists of four key components,

- **User groups** users in the online social community are organised into user groups with diverse content interest. Users inside each group collaborate via privacy preserving approaches such as homomorphic encryption and secret sharing, to protect user privacy from being inferred to recommended server. A host user of each group invites his/her friends to form a user group.
- **Interest groups** Inside each user group, interest groups are formed to identify true interests of users. Interest group identification ensures that users receive no “uninterested” items during recommendation process. In this system model, k-centroid clustering algorithm is adopted to identify the interest groups which clusters similar items to form groups. Interest groups also helps to generate pseudo users in user groups.
- **Pseudo users** Users in a user group maintain a set of pseudo users to interact with the recommender server to obtain recommendations on behalf of real users. Each pseudo user in the user group is associated to the relevant interest group to obtain appropriate recommendations. The server makes recommendations to the pseudo users based on their interests and users in the group re-calculates their personalized recommendations based on the importance of recommended items to them.
- **Recommendation algorithm** to make recommendations, the server first needs to collect users’ itemLists(interests of users). The secure transmission of users’ itemLists is achieved through efficient privacy preserving cryptography approaches such as Homomorphic encryption and secret sharing schemes. The combined itemLists of users in the social community allows server to perform proposed frequent itemset mining algorithm (Apriori algorithm) to generate association rules and make recommendations to the pseudo users. The recommendations made to the pseudo users are used by real users to calculate their own personalized recommendations.

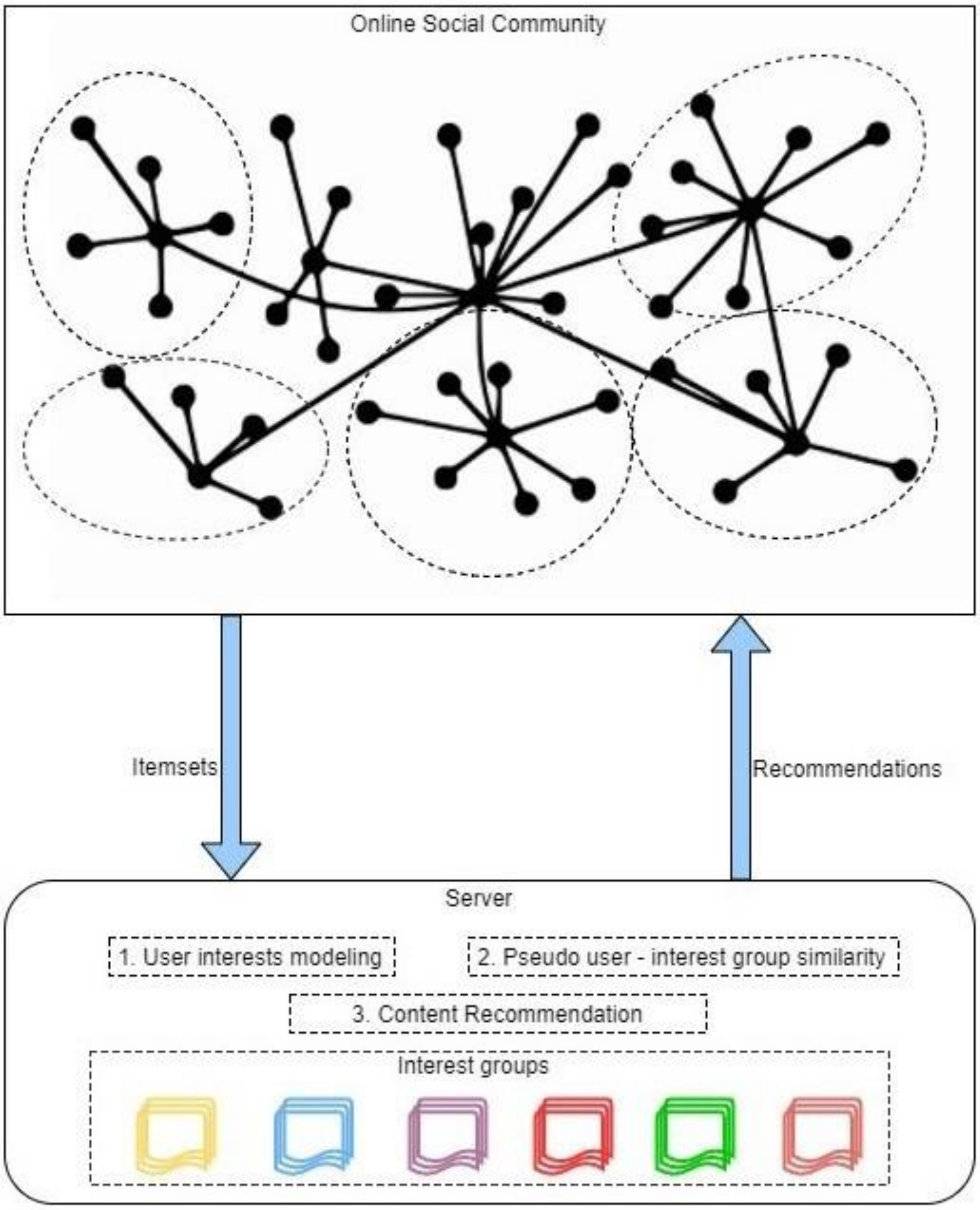

Online Social Community
Itemsets
Recommendations
Server
1. User interests modeling
2. Pseudo user - interest group similarity
3. Content Recommendation
Interest groups

Figure 1: System model of a Privacy-preserving content recommender for online social communities

4.2 User group definition and construction:

This section, describes how user groups are organized in a privacy-preserving manner. Within a given User group, users collaborate and transmit itemLists to server without sacrificing the privacy of any individual user.

Definition 1: A user group $\boldsymbol{g}$ is a three tuple: $\{u_g, I_g, p_g\}$ where $g \in G$ in which u_g is a set of users who collaborate to form a user group and hide privacy of each other, I_g is the set of Interest groups where each I_g in g contains items of similar content. p_g is a set of pseudo users who communicate with the server on behalf of the real users in $\boldsymbol{g}$ to deliver recommendations.

User groups can be formed in privacy-preserving fashion to hide contents of each user from a set of users participating in the same group. A group construction protocol is proposed which can automatically organize users into groups with diverse interests in a distributed and privacy-preserving fashion. User group construction is shown in **Algorithm 1** and for each user group *g* constructed, the number of users in *g* — K_g should be no less than 3. If g only contains two users (u_1 and u_2), then u_1*'s* privacy could be easily inferred from their jointly computation results by *u2*, and vice versa. For a user group g with $K \geq 3$, users' privacy can be protected in *g*. The formal proof is shown in section 4.6. The user groups are constructed in a peer-to-peer way and therefore should be noted that users may choose to leave a user group or join other group for various reasons. Thus, once a user requests to leave, the other users in the same user group should check whether their requirements are met. If all their requirements are still met, then the user group does not need to be changed. Otherwise, they should re-construct new user group using the above *SecureGrouping* algorithm by dismissing the previous group. The construction of user group is shown below in **Algorithm 1**.

Algorithm 1. SecureGrouping(U).

```
Require: $U$ is the set of users in the Social community.
1: For each user $u \in U$, $Su$ is the expected user group size of $u$;
2: while Not all users in $U$ are in user groups do
3:     for each $u \in U$ who has not joined any user group do
4:         u chooses to be the "host" of a user group with probability
$Pr_{host}(u) = S_u/|U|$
5:         if $u$ is host then
6:             $u$ invites its friends to join its group;
7:         end if
8:     end for
9:     for each do
10:        Let $Hu$ be the set of u's friends who are hosts of user
           groups;
11:    if $Hu \neq \emptyset$ then
12:        u randomly chooses $v \in Hu$ and joins the group of $v$;
13:    else
14:        Let $Ju$ be the set of u's friends who already joined user
           groups;
15:        if $Ju \neq \emptyset$ then
16:            $u$ randomly chooses $v \in Ju$ and joins the group of $v$;
17:        end if
18:      end if
19:    end for
20:    for each user group $g$ do
21:       Let $u$ be the host of $g$;
22:       if $|u_g| \geq Ku$ then
23:          $g$ is formed;
24:       end if
25: end for
```

```
26: end while
```

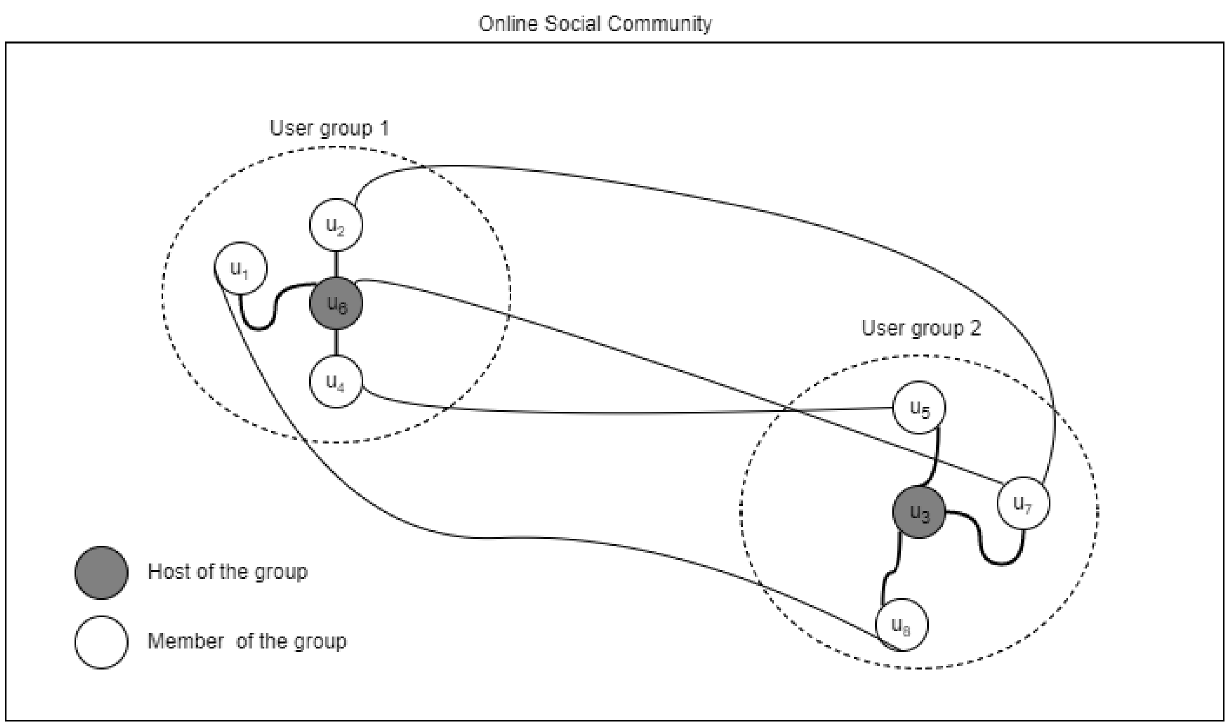


Figure 2. User grouping in online social communities

For example, let U = {u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8} be the users in the social community. A user from the social community chooses to be the host of group with the probability $Pr_{host}(u) = K_u/|U|$. If a user u from U is host then, he/she invites his/her friends to join the group. In this example, let users u_3 and u_6 be the hosts of the group and invites their friends to join their group. u_6 invites his set of friends $u_g = \{u_1, u_2, u_4\}$ to form a user group g_1 ***and*** u_3 invites his set of friends $u_g = \{u_5, u_7, u_8\}$ to form a user group g_2**.** **Figure 2** illustrates, user group formation.

In the next sections, we discuss how interest groups are formed, how pseudo users are generated and how the generated pseudo users are delegated to interest groups.

4.3 Users Interest modelling: Interest groups definition and construction:

This section describes how interest groups are formed in a privacy-preserving fashion.

Definition 2: A set of interest groups $I_g = \{I_{g1}, I_{g2}, I_{g3} .... I_{gk}\}$, where k is the number of interest groups, in which $I_g = \{i_1, i_2, i_3, ..., i_m\}$ is a set of items and c_g belongs to I_g is the centre of the group and represents "interest" of Ig and holds the property, for any two interest groups I_{gi} and I_{gj}, where $i \geq 1, j \leq k$ and $I \neq j$, $I_{gi} \cap I_{gj} = \emptyset$.

In this model, user interest modelling is performed by privacy-preserving user interest clustering algorithm, which clusters similar items into interest groups. After interest group modelling, each user group will have interest groups distribution to generate pseudo users. Here k-centroid clustering method is adopted to cluster similar items. The work flow of k-centroid clustering approach is described in **figure 3**,

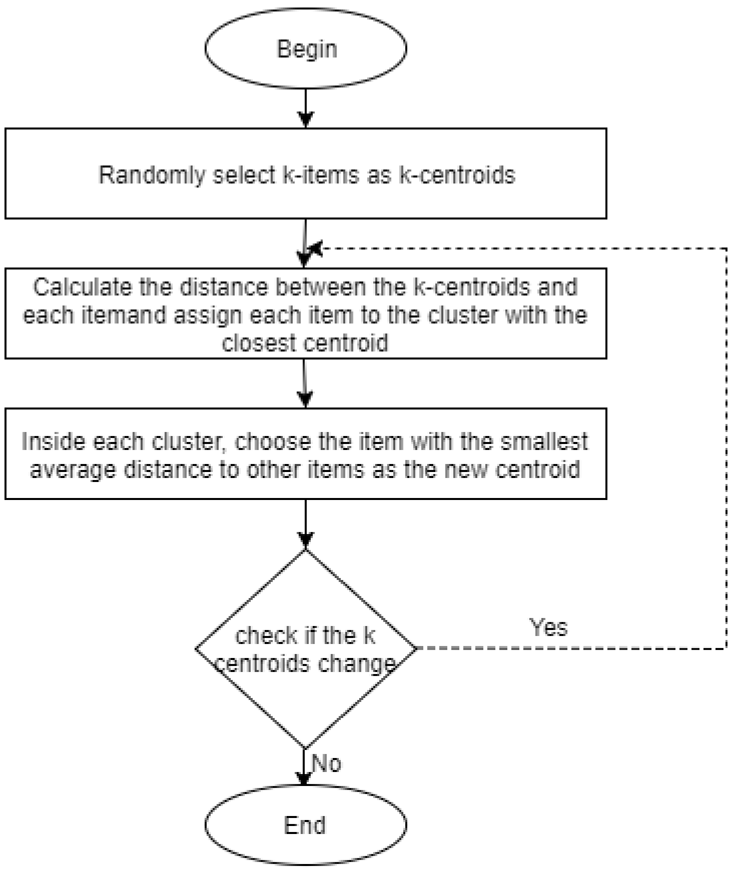

Begin
Randomly select k-items as k-centroids
Calculate the distance between the k-centroids and each itemand assign each item to the cluster with the closest centroid
Inside each cluster, choose the item with the smallest average distance to other items as the new centroid
check if the k centroids change
Yes
No
End

Figure 3. Workflow of k-centroid algorithm

Challenges in identifying interest groups:

- Optimal number of interest groups i.e. good inter-group separation and intra-group similarity. A better number of interest groups helps to generate accurate recommendations to users
- Privacy-preserving item similarity computation

4.3.1 Privacy-preserving item distance calculation:

Another challenge in the k-centroids clustering process is to compute the distance between items efficiently without compromising user privacy. To preserve user privacy, two communication protocols are proposed. First protocol is based on Elliptic curve cryptography and second one is based on Shamir's secret sharing scheme. The proposed communication protocol using the two approaches is illustrated in the figure 2. Once server receives the itemLists of all the users in the social community, it can perform item distance calculation based on Jaccard similarity. It has the following property,

$$JaccardSimilarty\ (i_1, i_2) = \frac{|i1 \cap i2|}{|i1 U\ i2|} \quad (1),$$

Which compares the similarity of the sets i_1 and i_2.

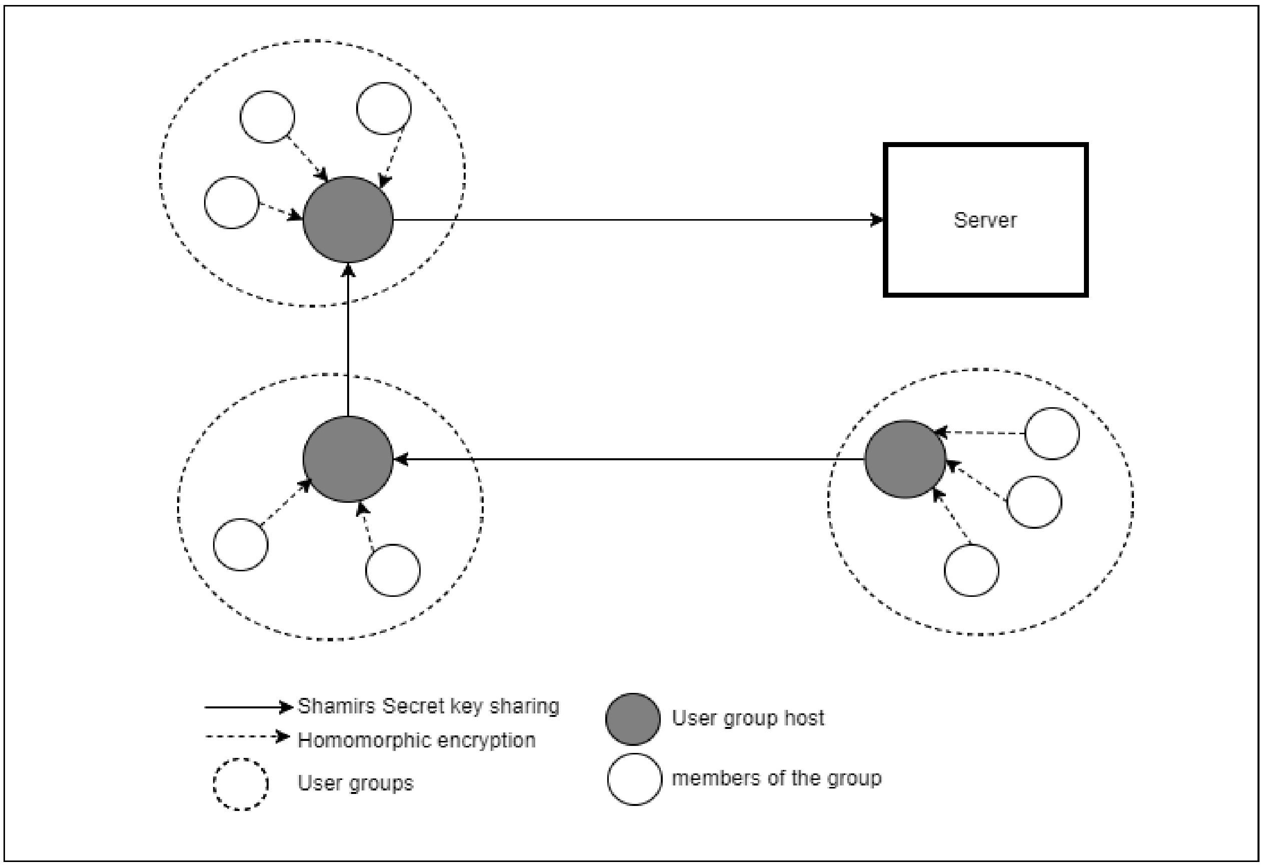


Figure 4. Proposed communication protocol

Phase 1: proposed protocol based on Elliptic-curve cryptography

Elliptic-curve-based Paillier public key cryptosystem is used in this phase as it requires shorter key length compared to RSA and Diffie-Hellman systems and saves significant computation time and memory space. The messages are signed with the help of secret key of Elliptic-curve-based Paillier public key cryptosystem before sending it to hosts of the group. This helps in validating the integrity and authenticity of a message.

- Certificate Authority (CA) generates public and secret keys to users, hosts and server. CA distributes public keys of all users to other users and private key to respective users and a share of secret key of server to user group hosts and server.
- Each user in the group computes local interests I.e. local itemLists and later each user encrypts the itemLists with public key of server and signs the encrypted itemLists with its own secret key. This encrypted and signed message is sent to the host of the group to which user belongs.
- Each host receives all the signed itemLists from all the users in the group, and later verifies the integrity and authenticity of signed message through respective public key of users. It shuffles and combines the received itemLists with its own encrypted itemLists, signs the combined itemLists through its own secret key and later sends it to the predefined host.

Phase 2: proposed protocol based on Shamir's secret key sharing
To prevent the collusion between host and server, Shamir's secret sharing scheme is used. As discussed earlier, the certificate authority distributes the public keys of each site to all other sites and distributes the secret key to respective hosts except the server. For reconstruction of secret key, server needs shares from all the hosts. Once, server reconstructs its secret key, it can decrypt itemLists of users which are signed by public key of server by all users, without revealing individual user itemLists. It generates a polynomial, in which constant term will be the secret key of server. Then it generates different shares of the secret key of server and distributes them to respective hosts. Now each host has one share of secret key of server. In phase 1, if server and host become malicious then they can collude with each other to reveal the itemLists of users. This is prevented using Shamir's secret sharing scheme since server cannot decrypt the itemLists until it has shares from all sites. For reconstructing the key, miner site needs shares from all sites, then it can decrypt the message. Thus, this approach prevents collusion of server and host.

The proposed protocol 2 works as follows: From the figure 2, u_6 and u_3 are hosts of user group 1 and user group 2, which have itemLists of the users $u_1, u_2, u_4, u_5, u_7, u_8$ along with itemLists of hosts of group u_6 and u_3, respectively. each host generates a polynomial of degree K. The hosts also agree on distinct random values vector $X = (x_1, x_2, \dots, x_n)$. Each host U_i chooses a random polynomial $p_i(x)$ of degree k, where $p_i(x) = I_i$ and $k = n\text{-}1$. Now, each host computes the shares of other hosts, including itself. Suppose host u_6 computes the shares, including itself as, share $(I_6, u_6) = p_6(x)$. Each host send these shares to respective predefined hosts as share (I_6, u_6), here in this case to host u_3. Now host u_3 gets the share $p_6(x)$ and add the received share to compute $T(x) = p_6(x) + p_6(x)$. The result is sent to the server as the host U_3 is last host. Thus, each host computes the global itemLists without revealing the local itemLists of real users. Once the server decrypts the global itemLists, it can estimate item distances based on Jaccard similarity between two sets.

After calculating item distances based on the Jaccard similarity method, the server can cluster all the items into interest groups with different cluster numbers - k and, choose the optimal k by the BIC score-based method. Then, the clustering with optimal k is the optimal interest groups clustering. These interest groups can help increase recommendation accuracy and generate pseudo users inside each user group. From previous example, let U be the user group in the social community and $U = \{u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8\}$ be the users in the user group U. Let interests of each user in the user group be $u_1 = \{i_1, i_2\}$, $u_2 = \{i_3, i_4\}$, $u_3 = \{i_2, i_4\}$, $u_4 = \{i_1, i_5, i_6\}$, $u_5 = \{i_7\}$, $u_6 = \{i_1, i_2, i_5\}$, $u_7 = \{i_5, i_6\}$, $u_8 = \{i_2, i_8, i_9\}$. As discussed, Server receives the itemLists of the users in social community securely through prposed communication protocol 1 and 2. Then, server estimates the distance between items using Jaccard similarity by building standard user-item matrix. After estimating distances between the items, k-centroid algorithm is performed to cluster similar items. Assuming that item-distance and clustering is performed on users items, derived interest groups could be $I_{g1} = \{i_1, i_2, i_4, i_7\}$ and $I_{g2} = \{i_3, i_5, i_6, i_8, i_9\}$.

4.4 Pseudo user management:

After interest grouping, pseudo users are generated to protect user privacy during recommendation process. As discussed earlier, Server interacts with the real users through the pseudo users, recommendations from server are sent to respective pseudo user, so that the server cannot obtain the personal information of individual real users. Each pseudo user acts as a “delegate” for an interest group, and the recommendations that the server makes to the pseudo user can be utilized to generate personalized recommendations by real users who have that interest.

4.4.1 Pseudo users generation:

Pseudo users are generated inside each user group, based on the interest groups calculated in the previous section. Each user group obtains set of interest groups calculated by the server. Then, users in the same user group constructs a set of pseudo users, each of which “delegates” a unique interest group. For each interest group delegated to the pseudo user inside a group, pseudo user profile is maintained, which is nothing but the itemLists of interest group associated. Pseudo user profile is required for the users in group to associate themselves to interest group for receiving recommendations efficiently.

4.2.2. Delegation of pseudo users to interest groups:

Given a set of interest groups, our goal is to associate pseudo users to the corresponding interest group. Algorithm, 2 illustrates how pseudo users are generated in a given user group.

Algorithm 2. SecurePseudoUSer(*g*, *Ig*)**.**

Require: *Ug* is the set of users in a given user group *g*, *Ig* is set of interest groups.

```
1: $P_s = \emptyset$;
2: For each user $u \in u_g$, $g_u$ is the expected user group size of $u$;
3: while Not any user in $U_g$ are pseudo users do
4:     for each $u \in u_g$ who is not a pseudo user do
5:         for each interest group $i_g \in I_g$ do
6:             u chooses to be the “pseudo user” of a user group with probability
$Pr_{pseudo}(u) = g_u / |U_g|$
7:             if $u$ is pseudo user then
8:                 chooses another user in the group to be a pseudo user;
9:             end if
10:                Assign user u as pseudo user for the interest group
11:                $P_s = \{u\}$;
12:        end for
13:    end for
14: return: $P_s$;
```

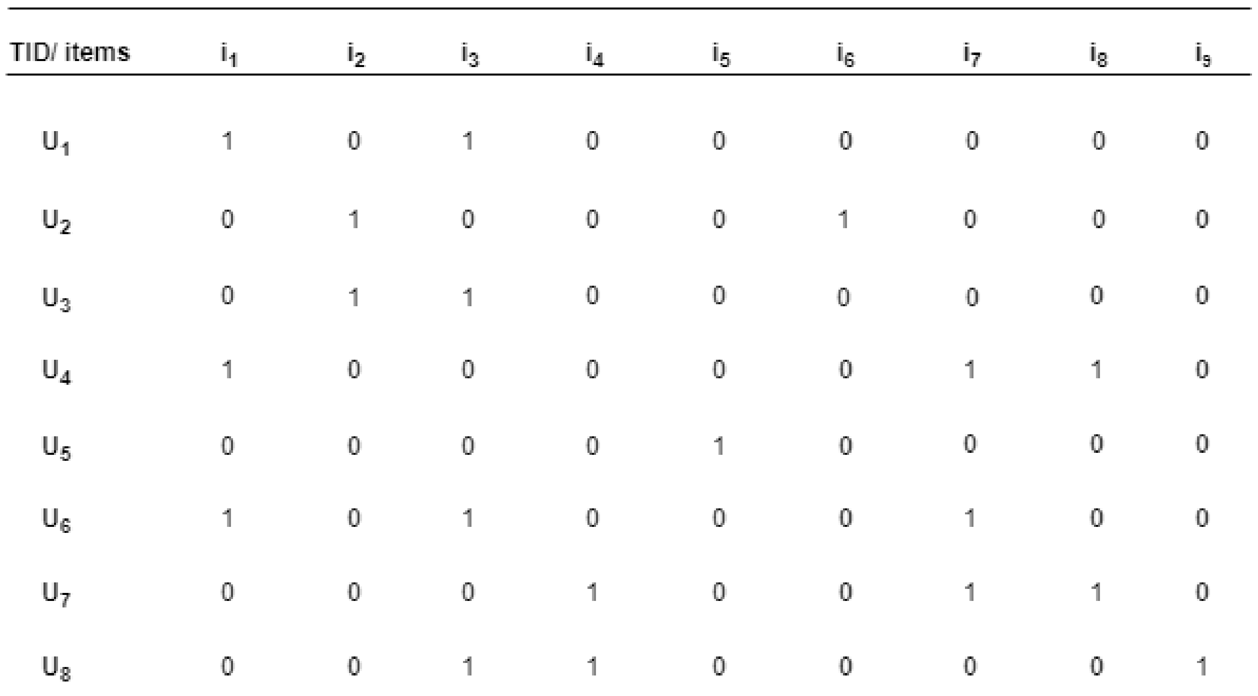
4.5 Content Recommendations

After user grouping and generation of pseudo users in each user group, the server can collect the itemLists of all users which is achieved from the previous section. Then, the server can make recommendations to the pseudo users based on the association rule mining approach to perform frequent itemsets mining.

4.5.1 Server-side recommendation. Server requires standard user-item rating matrix, which is required in association rule mining to generate association rules which are considered as recommendations. From the section 4.3.1, it is known that server receives itemLists from all the users securely through communication protocols 1 and 2. Therefore, server can construct user-item rating matrix without knowledge to which user the itemList belongs.

For the recommender server, Apriori algorithm is employed for discovering association rules to recommend items to users. In this system, the server cannot see real user data, but only global user data which is privacy-preserving as no real user data is revealed. Thus, the server will calculate the association rules from the obtained matrix of users-items and makes recommendations to each pseudo user associated with an interest group. As discussed in chapter 2, association rule mining is described as,

An association rule is an implication in the form of $A \Rightarrow B$, where $A, B \subset I$ are sets of items called itemsets, T be transaction that contains a set of items such that $T \subseteq I$, D be a database with different transactions T_s and $A \cap B = \emptyset$. A is called antecedent while B is called consequent, the rule means A implies B. The basic measures of association rules are support(s) and confidence(c). Association rule mining is to find out association rules that achieve minimum support count and confidence from a given database.



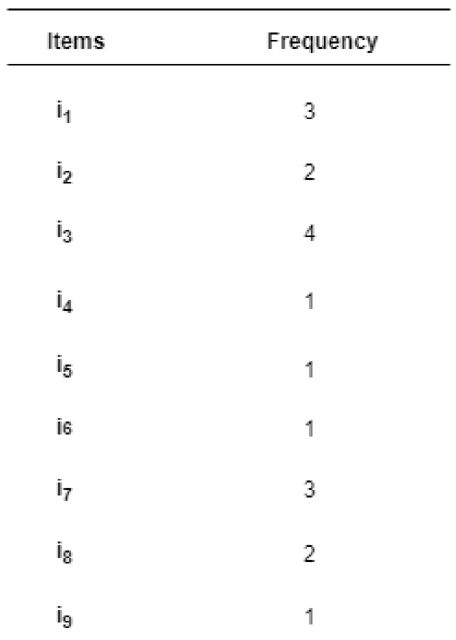
TID/ items	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8	i_9
U_1	1	0	1	0	0	0	0	0	0
U_2	0	1	0	0	0	1	0	0	0
U_3	0	1	1	0	0	0	0	0	0
U_4	1	0	0	0	0	0	1	1	0
U_5	0	0	0	0	1	0	0	0	0
U_6	1	0	1	0	0	0	1	0	0
U_7	0	0	0	1	0	0	1	1	0
U_8	0	0	1	1	0	0	0	0	1

Table 1. User-item rating matrix

From previous example, let g be the user group in the social community and $U = \{u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8\}$ be the users in the social community. Assuming, interests of each user be $u_1 = \{i_1, i_2\}$, $u_2 = \{i_3, i_4\}$, $u_3 = \{i_2, i_4\}$, $u_4 = \{i_1, i_5, i_6\}$, $u_5 = \{i_7\}$, $u_6 = \{i_1, i_2, i_5\}$, $u_7 = \{i_5, i_6\}$, $u_8 = \{i_2, i_8, i_9\}$. Server receives the itemLists of the users in social community securely through employed communication protocol 1 and 2 from previous section. Table 1, shows the user-item matrix where '1' denotes that user likes the item and '0' denotes user is not interested in an item. However, each row in the user–item matrix does not belong to real user in social community. Therefore, transactions of user likes are built. As discussed, earlier server has no knowledge of real users itemLists.

Server after generating user-item matrix, performs Apriori algorithm to generate the association rules for the content recommendation. In the above example, let the minimum support threshold be 2 and length of association rules generated be 2. Server receives a frequency all the items that occur in all the transactions.

Step1: Server then creates a candidate itemsets table for all the items along with the support count. In this case, Table 2. Shows candidate itemsets C_1 are generated then transactions from table 1. are scanned to check support count of corresponding itemsets



Items	Frequency
i_1	3
i_2	2
i_3	4
i_4	1
i_5	1
i_6	1
i_7	3
i_8	2
i_9	1

Table 2. Candidate itemsets C_1

Step 2: We know that only those elements are significant for which the support is greater than or equal to the minimum support threshold. Here, support threshold is 2 as discussed above, hence only those items are significant which occur in more than one transaction and such items are i_1, i_2, i_3, i_7, i_8. Table 3. Shows Large-1 L_1 itemsets generated by pruning those itemsets whose support counts are below the pre-defined threshold.

Items	Frequency
i_1	3
i_2	2
i_3	4
i_7	3
i_8	2

Table.3 Large-1 L_1 itemsets

The table. 3 above represents items liked by users frequently.

Step 3: The next step is to make all the possible pairs of the significant items keeping in mind that the order doesn't matter, i.e., AB is same as BA. To do this, take the first item and pair it with all the others such as i_1i_2, i_1i_3, i_1i_7, i_1i_8. Similarly, consider the second item and pair it with preceding items, i.e., i_2i_3, i_2i_7, i_2i_8 and continue for third and fourth item i_3i_7, i_3i_8, i_7i_8. So, all the pairs in this example are i_1i_2, i_1i_3, i_1i_7, i_1i_8, i_2i_3, i_2i_7, i_2i_8, i_3i_7, i_3i_8, i_7i_8.

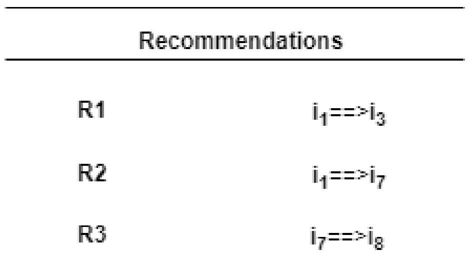
Step 4: Algorithm counts the occurrences of each pair in all the transactions and only those itemLists are significant which cross the minimum support threshold and those are,

Itemsets	Frequency
i_1i_3	2
i_1i_7	2
i_7i_8	2

Table 4. Large-2 L_2 itemsets

Therefore, the set of 2 items that were liked most frequently are i_1i_3, i_1i_7, i_7i_8.

Step 5: Till now, server performed the Apriori algorithm with respect to frequent itemset generation. Next task is to find the association rules efficiently. In this example, I will not go deeper into the theory of the Apriori algorithm for rule generation. As discussed earlier the maximum length of association rule generated is gives as 2. We find that i_1i_3, i_1i_7, i_7i_8 are the frequent itemLists that support the minimum support threshold and maximum length. Therefore, possible association rules *are* $i_1 \rightarrow i_3$, $i_1 \rightarrow i_7$, $i_7 \rightarrow i_8$. Which indicates users who are interested in item i_1 are also interested in item i_3 and so on for other two association rules derived. Therefore, the server recommendations are shown in table 5,



Recommendations	
R1	$i_1 ==> i_3$
R2	$i_1 ==> i_7$
R3	$i_7 ==> i_8$

Table 5. Association rules for recommendation

4.5.2 Client-side recommendations. The item recommendation from the server can only reflect how the pseudo users like the items. After obtaining the recommendations, real users should re-calculate their personalized recommendations of items based on the server-side recommended items. As discussed, server-side recommendations contain antecedent and consequent. For instance, $i_1 \rightarrow i_3$ where i_1 is called antecedent while i_3 is called consequent, the rule means i_1 implies i_3. This association rule generated is passed as recommendation to the real users through pseudo users. A real user verifies if antecedent is present in his itemList. If present, user gets the consequent as recommendation. On the other hand, if a real user does not have the antecedent in his itemList, the recommendation is not received. Rating of a recommendation (item) is calculated as,

$$Rating\ (r_i) = |I_u \cap I_{r,antecedent}| \quad (2),$$

Where I_u is set of items user likes $I_{r,antecedent}$ is set of items recommended to user. If $r_i > 0$, user receives the recommendation. Otherwise, the recommendation is not received by a real user.

For example, figure shows the recommendation process in user groups 1 and 2. Recommendation *R1* and *R2* is received by respective pseudo users u_2 and u_7 in the user groups 1 and 2. Each real user in the group calculates r_i of the items recommended with pseudo user with the equation 2. Here in this example, *R1* and *R2* are received by all users in the group 1 and $r_i > 0$, while *R3* is not received by any user in the group 1 as $r_i = 0$. In group 2, *R1*, *R2*and *R3* are not received by any user as $r_i = 0$.

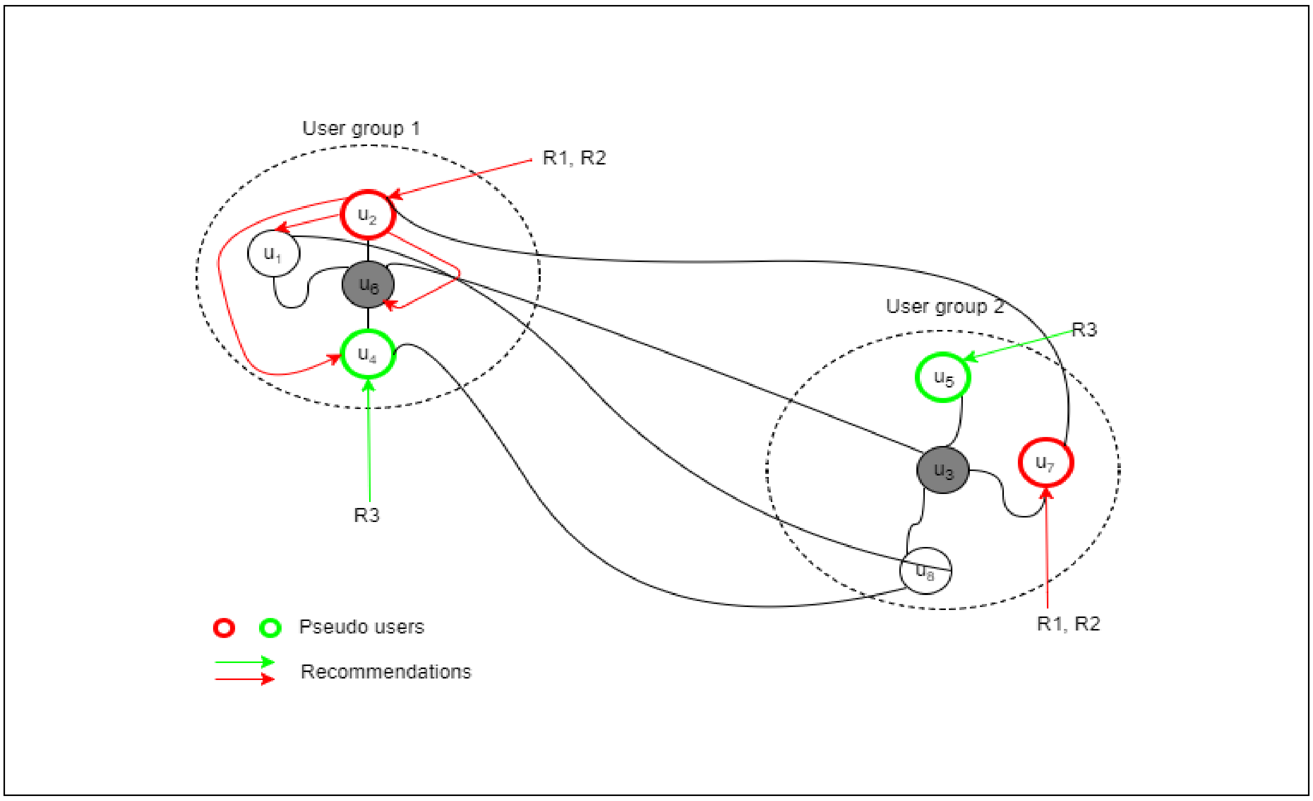


Figure 5. Example client-side Recommendation

4.6 Privacy preserving in system model:
In the previous sections, technical details of the working model have been presented. It is seen that users privacy is preserved by group of users collaborating to hide individual users privacy. In In this section, we discuss the privacy preservation feature of this model. First, it is shown that the proposed user group structure can protect user privacy. Then, it is proved through theorems that user privacy will not be exposed by computations inside user groups, which are user group construction, user interest modelling, pseudo user management, and content recommendation.

4.6.1 Privacy preserving in user groups:
Theorem 1. Let U be a set of users in an online social community ($|U|> 1$). The execution of SecureGrouping (**Algorithm 1**) on U reveals none of the interest privacy of users in U.

Proof. During the execution of SecureGrouping, the only step that contains communication is to choose a user group to join, in which no information exchange takes place. Therefore, the SecureGrouping protocol will not expose user privacy at all.

Theorem 2. Let g be a user group constructed by **Algorithm 1**, $g\geq 3$. Π(u1,u2) is an algorithm for user u1 to infer the interest privacy of user $u2$ in the same user group during the recommendation process (following the semi - honest behaviour). Let $\Pi(u1,u2\ )$ denote the result of $u1$ executing Π on $u2$. Then, for any user $u \in g$ and any other two users, u1u2∈g, $\Pi(u,u1\ )$ and $\Pi(u,u2\ )$ are perfectly indistinguishable.

Proof. Let $\{j\in I|\ \exists\, p\in pg,\ r_i > 0\}$ be the set of items satisfying $r_i > 0$ in g, where I is the set of items in the system. The only information that can be utilized by Π are u's input data Input u and the computation outputs Output, because all the other intermediate data are encrypted using communication protocols 1 and 2, in this system and no further information could be obtained by u as users in g are not colluding in the semi-honest model.

Then, for each item $i \in \{j \in I | \exists p \in pg, r_i > 0\}$, the information about i is contained in $Output(i) - Input(i)$, where $Output(i)$ is the output of the computation on i and $Input(i)$ is the input of u in the the computation on i. There are two scenarios to be discussed:

- $Output(i) - Input(i) = 0$. In this case, neither u_1 nor u_2 is interested in i, so that $Pr(i \in \Pi(u,u1)) = 0, Pr(i \in \Pi(u,u2)) = 0$
- $Output(i) - Input(i) > 0$. As $Output(i) - Input(i)$ is the combined information of u_1, u_2 and users in $g - \{u, u1, u2\}$, so that $Input(i)$ and $Input_{u2}(i)$ cannot be distinguished from $Output(i) - Input(i)$ in the semi-honest model where no users are colluding, i.e., $Pr(Input_{u1}(i)) > 0) = Pr(Input_{u2}(i)) > 0)$

In both cases, we have $Pr(i \in \Pi(u,u1)) = Pr(i \in \Pi(u,u2))$

Thus, we can say that $\Pi(u, u1)$ and $\Pi(u,u2)$ have the same distribution over $\{j \in I | \exists p \in pg, r_i > 0\}$.i.e that $\Pi(u,u1)$ and $\Pi(u,u2)$ are perfectly indistinguishable.

4.6.2 Privacy protection in user interest modelling:

In this model, user interests are modelled by the proposed user interest modelling algorithm. Here, we prove that the proposed user interest modelling method is privacy-preserving for users and protects users interest from adversary attacks. The first protocol is based on Elliptic-curve cryptosystem, while the second protocol is based on Shamir's secret sharing scheme.

Theorem 3. Let U be a set of users in an online social community ($|U| > 1$). The execution of the proposed user interest modelling method on U is privacy - preserving for all users in U.

Proof. During the user interest modelling process, communication protocol 1 and protocol 2 are the two steps required for item similarity computation. All other steps are performed by the recommender server, during which no user interest privacy could be obtained. By applying the Theorem, if we can prove that communication protocol 1 and communication protocol 2 is privacy-preserving, then we can prove that the user interest modelling method is privacy-preserving. Here, communication protocol 1 and protocol 2 are discussed in two phases and prove they are privacy preserving:

Phase 1: In this stage, each user u in user group sends his item list to the host of the group. If u chooses not to add its itemList then output of u is empty list. Therefore, no itemList is sent to the host of group. In this protocol, Elliptic-curve-based Paillier public key cryptosystem since it requires shorter key length and provides the same level of security as discussed earlier. Each user u signs the itemList with the help of Elliptic-curve-based Paillier public key cryptosystem before sending it to host which in turn helps in validating the integrity and authenticity of itemList sent by each user in the system. Host of the group receives all the signed itemLists from all the users in the group and verifies the integrity and authenticity of signed itemLists through repective public keys of users. Also, host shuffles and combines the received itemLists with its own signed itemList. Later combines the signed itemLists of all the users in the group with its own secret key and send it to the next host. Here, certificate authority does not have any database part and generates the Elliptic-curve based Paillier public and secret keys for all the involving users in social community. we see that the homomorphic property of Elliptic-curve-based Paillier cryptosystem helps find the itemLists of all the users in the group securely. The proposed communication protocol works as follows:

For an each itemList *I* that belongs to *(n - 1)* users, the users itemLists in the group can be derived as follows.

Encryption: $E(I_1 + I_2 + I_3 + I_4 + \ldots + I_{n-1}) = E(I_1) * E(I_2) * E(I_3) * E(I_4) * \ldots * E(I_{n-1})$

Decryption: $D(E(I_1 + I_2 + I_3 + I_4 + \ldots + I_{n-1})) = I_1 + I_2 + I_3 + I_4 + \ldots + I_{n-1}$

After the decryption process, the result will be equal to combined itemLists *I* of all the *(n-1)* users in the user group.

The proposed communication protocol 1 securely collects the itemLists of the users in a group, since all the information is performed after performing encryption and signing. Also, ensures integrity and authencity of the received information. However, the itemLists are not sent to server as it may fail if host of a group colludes with the server. There Shamir's secret sharing scheme is employed to prevent collusion.

Phase 2: The proposed Shamir's secret sharing scheme helps to prevent collusion between host and server. The certificate authority distributes the public keys of each host to all other hosts and distributes the secret key to respective host except the server. It generates a polynomial, in which constant term will be the secret key of miner site. Then it generates different shares of the secret key of server and distributes them to respective hosts. Now each host has one share of secret key of miner site. In protocol 1, if server and host become malicious then they can collude with each other to reveal the itemLists of other users in the system. This is prevented using Shamir's secret sharing scheme since server cannot decrypt the itemLists until it has shares from all hosts. For reconstructing the key, server needs shares from all hosts, then it can decrypt the itemLists of all the users. Through this approach, collusion of hosts and server can be prevented. The proposed communication protocol works as follows:

Consider three hosts *H1, H2 and H3,* where each host holds itemLists *I* as I_1, I_2 *and* I_3, respectively. Now each site wants to compute $I = I_1 + I_2 + I_3$ without revealing their local ItemLists to each other. Each host computes shares of secret key as $share(I_1, H_1) = p_1(x)$, $share(I_2, H_2) = p_2(x)$, $share(I_3, H_3) = p_3(x)$. The last host interacting with server gets all the shares and adds all the received shares to compute $T(x) = p_1(x)+p_2(x)+p_3(x)$ and sends this result to server. Then server can decrypt the global itemLists and does not reveal individual users privacy.

Thus, the communication protocol 1 and communication protocol 2 are privacy-preserving for all users. Then, by applying the Theorem, we can say that the proposed user interest modelling method is privacy-preserving for all users in U.

4.6.3 Privacy protection in content recommendation:

User ItemLists for recommendation: Association rule mining is achieved by the proposed communication protocol 1 and communication protocol 2. Privacy preservation feature of proposed protocols are discussed in Section 7.4, which shows that user privacy can be protected against group members. And all itemLists are sent to the server via host, so that the recommender server can know nothing about the privacy of real users.

Server-side recommendation: the server-side recommendation is only using the interest profiles of pseudo users, so that the privacy of real users is protected from the server. Meanwhile, we have shown in …that user privacy can be protected inside each user group when maintaining pseudo users. Thus, the server-side recommendation is privacy-preserving.

Client-side recommendation: the client-side recommendations are all computed based on recommendation of pseudo users profiles. Recommendation scores to pseudo users contains no privacy of real user, and user interest distributions are stored locally on. Thus, user privacy would not be exposed in client-side recommendation.