Int. J. Inf. Secur. (2016) 15:335–360
DOI 10.1007/s10207-015-0296-7

REGULAR CONTRIBUTION

A practical privacy-preserving targeted advertising scheme for IPTV users

Leyli Javid Khayati[1] · **Cengiz Orencik**[1] · **Erkay Savas**[1] · **Berkant Ustaoglu**[2]

Published online: 26 July 2015

Abstract In this work, we present a privacy-preserving scheme for targeted advertising via the Internet Protocol TV (IPTV). The scheme uses a communication model involving a collection of subscribers, a content provider (IPTV), advertisers and a semi-trusted server. To target potential customers, the advertiser can utilize not only demographic information of subscribers, but also their watching habits. The latter includes watching history, preferences for IPTV content and watching rate, which are periodically (e.g., weekly) published on a semi-trusted server (e.g., cloud server) along with anonymized demographics. Since the published data may leak sensitive information about subscribers, it is safeguarded using cryptographic techniques in addition to the anonymization of demographics. The techniques used by the advertiser, which can be manifested in its queries to the server, are considered (trade) secrets and therefore are protected as well. The server is oblivious to the published data and the queries of the advertiser as well as its own responses to these queries. Only a legitimate advertiser, endorsed with so-called *trapdoors* by the IPTV, can query the cloud server and access the query results. Even when some background information about users is available, query responses do not leak sensitive information about the IPTV users. The performance of the proposed scheme is evaluated with experiments, which show that the scheme is practical. The algorithms demonstrate both weak and strong scaling property and take advantage of high level of parallelism. The scheme can also be applied as a recommendation system.

Keywords IPTV · Targeted advertising · Privacy · Cryptography · Cloud computing

1 Introduction

The literature suggests [2] that targeting content (e.g., advertisement, online content) to prospective customers is already a huge and lucrative business. Traditional media such as TV, radio or newspaper can do only a little to customize advertisements of products or services for their customers. It is claimed [3] that producers are more willing to send advertisements of their goods to prospective customers with high accuracy. By making use of online history, observed behaviors and demographics, online media allows finer customer targeting. Therefore, marketing based on online traits and demographics is preferred by advertising agencies aiming to increase the benefit from advertisements. However, a major issue is the potential violation of the privacy of individuals.

IPTV is an online media with potentially millions of subscribers, and thus, it is a hot spot for advertising agencies that have the incentives to analyze the data available to IPTV. For example, changing a channel requires sending a request to the IPTV provider, which allows for accurately tracking subscribers' watching habits. This information can be

A preliminary version of this work has been presented at Secrypt 2012 [1].

✉ Cengiz Orencik
cengizo@sabanciuniv.edu

Leyli Javid Khayati
leyli@sabanciuniv.edu

Erkay Savas
erkays@sabanciuniv.edu

Berkant Ustaoglu
berkantustaoglu@iyte.edu.tr

[1] Faculty of Science and Engineering, Sabanci University, Istanbul, Turkey

[2] Izmir Institute of Technology, 35430 Izmir, Turkey

an invaluable asset from advertising companies perspective. From another point of view, however, it is considered as a threat to subscribers' privacy [4]. Therefore, exploitation of such information for processing such as targeted advertising is not acceptable without proper security and privacy protection.

The data collected by the IPTV are accumulating over time and, therefore, there is an increasing motivation to outsource data warehousing to cloud services. Such an outsourcing can decrease IPTV expenses by mitigating the burdens of storage, service management and expenditure on hardware or software [5]. However, outsourcing amplifies the need for privacy protection since storing personal and identifiable data on such servers exacerbates privacy risks as identified in [6].

The media attention given to cloud computing suggests that it is gaining a considerable attention in business environments allowing for remote storage and data access, without much deliberation for day-to-day management. Sensitive information such as health records, emails and private photos can also be stored on cloud servers. Although data encryption prior to outsourcing is suggested to protect privacy and prevent unauthorized cloud access [7], processing encrypted data for secondary usage (e.g., targeted advertisement based on online traits and demographics) is a challenging task.

In our setting, the advertiser matches IPTV subscribers with particular advertisements by mining demographics and watching preferences data. Data mining operations are performed on a remote database of subscribers' data (e.g., kept in a cloud server) by sending queries. Utilizing the responses to the queries, the advertiser aims to match subscribers only with relevant advertisements. Queries have to be encrypted to prevent the disclosure of the strategy of the advertiser to external parties such as the cloud server or other advertisers. Thus, there is a need to query an encrypted database with encrypted keywords. The core challenge we address in this paper is facilitation of such operations, in the context of a practical privacy-preserving targeted advertisement scheme.

The setting here is different from most of the previous works in the literature [2,3], whereby the IPTV provider (i.e., controller of the database) is also in charge of data processing and advertisement selection. Naturally, in such context subscriber privacy and advertising strategy secrecy are not major concerns. Nevertheless, use of private data for secondary processing requires both the consent of the user and proper precautions such as anonymization and encryption. Indeed, those in charge of data management are not necessarily trusted, independent from whether they are IPTV employees or third parties. Therefore, one must assume that the server deployed to hold subscribers' data is not fully trusted and potentially the data or its management is outsourced to a so-called honest-but-curious third party, where the server does not modify the message content and flow, but may analyze them to infer additional information.

In our setting, an entity called IPTV collects data about its customers called subscribers, keeps the data on an external cloud server and authorizes designated advertisers to mine the data subject to not violating the customers' privacy. The authorized advertisers can send targeted advertisements to a subset of the subscribers that satisfies the constraints imposed by the advertiser. The constraints for selecting the target subset of subscribers are given via a set of keywords, and only those subscribers satisfying these constraints receive the advertisement. The sensitive information of subscribers is hidden both from the advertisers and from the cloud server. We also consider the case, in which some background information about the statistics of the subscribers is known by a malicious advertiser. The queries of honest advertisers may carry sensitive information about the advertiser strategy, and therefore, keeping the queries secret from the cloud server is another important aspect of the model discussed here.

The rest of this paper is organized as follows. In the next section (Sect. 2), we review the related works on searchable encryption schemes and secure targeted advertising. In Sect. 3, we outline the contributions of this work. Section 4 provides the entities involved and formalize the secure targeted advertisement problem. The preliminary information such as data filtering and data hiding techniques is provided in Sect. 5. The privacy requirements are formalized in Sect. 6. In Sect. 7, we provide a detailed description of the proposed scheme. Section 8 explains a key technique to provide stronger protection for the case, where adversary has some background knowledge on the model. The computation of collaborative filtering over encrypted data is explained in Sect. 9. In Sect. 10, we provide the security guarantees. In Sect. 11, we analyze the performance of the method using communication and computation costs. We discuss the results from our extensive set of experiments in Sect. 12. We discuss some future directions and conclude in Sect. 13.

2 Related work

The cloud server is not a fully trusted entity, and therefore, the subscribers' data must be anonymized prior to outsourcing. One particular anonymization technique suggests to cluster the subscribers that have close demographics and watching traits. Then the representatives of these clusters would be placed in the cloud server and the advertisers can use these summaries to match the relevant subscribers with their advertisement portfolio. The technique is similar to the k-

anonymity [8] algorithm, but hardly the best option in our application scenario for many reasons: (i) clustering leaks the information of some subscribers, (ii) the advertisers have to use static clusters formed by the IPTV, (iii) IPTV cannot control who access/utilize the data and (iv) loss of accuracy. Since leaking subscribers information is the most crucial problem from the privacy perspective, more robust solutions are needed.

Safeguarding private information requires that sensitive data (and associated searchable index file) be encrypted before outsourcing. However, data encryption hinders traditional data utilization techniques based on keyword search. Considering large number of users and documents, it is crucial for the search service to facilitate fast and efficient multi-keyword queries.[1]

Many searchable encryption schemes in the literature focus on single keyword search operation. In the public key cryptography setting, Boneh et al. [9] presented the first searchable encryption construction using public key cryptosystem to perform search on encrypted data. Later, Wang et al. [10] described a ranked keyword search method over encrypted cloud data. In this work, the server learns the relevance order of documents that contain a specific keyword, and it is only limited to single keyword search queries. Several works on multi-keyword search that enable conjunctive and disjunctive search options exist [11–14], but these schemes incur large overhead in computation and communication costs. A recent work by Cash et al. [15] introduces a searchable encryption scheme that supports multi-keyword search operation over encrypted database. While due to scalability the scheme is suitable for big data applications, it does not allow to rank the responses based on their relevancy to a given query.

Wang et al. [16] proposed an efficient scheme for conjunctive keyword-based search. In their solution, a searchable index is generated for each document that represents the keywords in those documents. Via suitable application of cryptographic hash functions, they obtain indexes and trapdoors that allow secure searching for keywords in the index file. However, the original scheme in [16] also does not cope with the problem of ranking the query responses in a relevancy order.

Our previous work [1], inspired from Wang et al. [16], utilizes a keyed hash function (HMAC) to map keywords in a subscriber's data to a sequence of r-bit indexes using a secret key known only to the IPTV. A similar approach exists in [17] for multi-keyword search over encrypted data. Advertisers must in advance obtain the so-called *secure trapdoors* from the IPTV, which allow searching the corresponding keywords within the subscribers' data. Since those trapdoors are generated using the IPTV's secret key, the server cannot learn the keywords in the advertiser's query. The proposed solution addresses relevant security and privacy requirements of interactive TV [18], which are also applicable in the scenario here. However, returning pseudonyms as done in [1] may disclose information about subscribers if the advertiser has background knowledge about the subscribers or if the cloud server knows the statistics of the data set. Further countermeasures are needed in that case. In the remainder, we propose one solution that addresses all those issues.

[1] Multi-keyword (or multi-predicate) queries are essential to find database entries that are relevant to all keywords in the query.

3 Contributions

Extending the basic scheme in [1], we provide an efficient solution that resists attacks even when some background knowledge is available. In the new scheme, the advertiser does not learn the pseudonyms of matching subscribers, who will ultimately receive a given advertisement. We also account for malicious advertisers that have background knowledge of the frequencies of the keywords in the data set. We adopt an aggregate data release method that protects the privacy of subscribers in the presence of background knowledge. Thus, the proposed method provides an efficient setting that facilitates data mining operations on a secure targeted advertisement application without violating the privacy of the IPTV subscribers and advertisers. We also enhance the formal definitions and provide an analysis showing that the proposed scheme satisfies these requirements.

We alleviate the risk of sensitive information leakage due to background information via the following countermeasures:

Anonymization: Subscriber's demographic information is anonymized, and therefore, re-identification of subscribers is difficult. Furthermore, the outsourced data set is obfuscated by introducing fake profiles, which eliminates the correlation of original data set statistics with the statistics of the obfuscated data set used in the cloud server.

Encryption: The demographic information (e.g., age, gender, occupation) about subscribers is encrypted via homomorphic encryption techniques [19] so that the cloud server can perform calculations directly over the encrypted data without decrypting it.

Aggregate data release: In case background information about the subscribers is available, the server returns only aggregate information on the set of matching subscriber profiles, in which sensitive information is sufficiently diverse. That is, the advertiser receives only aggregated demographic information for the matching

subscribers along with the total number of matches. If the number of matching results is sufficiently high, the matching data are not considered sensitive and the aggregated data are released. Otherwise, the aggregated result is not released except for the number of matching profiles. In the latter case, the advertiser still learns the number of recipients of a particular advertisement.

The benefits of the proposed scheme are multifold: (i) The IPTV is partially relieved of management cost and processing of data, (ii) advertiser's mining techniques are less exposed to the cloud server, (iii) data mining required for targeted advertising can be performed by advertising parties that have the relevant expertise and tools and (iv) the IPTV can generate additional revenue from subscribers' data by ensuring sufficient data protection safeguards in comply with relevant legislation, such as EC Directive 95/46 [2] [20]. The scheme can also be applied when the advertiser is in fact a subdivision within the IPTV (e.g., private cloud). In this case, the IPTV has robust data protection practices that can meet the relevant legislation, namely sensitive data are encrypted; data used for targeted advertisement include only necessary, anonymized information about the subscribers; and access control to data can be exercised in a fine-grained manner.

Another contribution of this work is the capability of secure *collaborative filtering*. Utilizing collaborative filtering, the IPTV can predict the preferences of the subscribers and use these predictions to send recommendations. Hence, the proposed scheme is not only an advertising scheme but also a recommendation system.

While the primitive adopted in our proposal is an efficient privacy-preserving keyword search, our main contribution is not a novel keyword search method, but a comprehensive solution for privacy-preserving targeted advertisement application in an IPTV setting.

Lastly, we implement and test the proposed scheme on both real and synthetic data sets, which show that the scheme is suitable for practical usage. We show that the proposed method provides a technique for advertisers to evaluate relevant prediction values about potential customers.

4 The problem statement

The proposed scheme aims to send targeted advertisements to the IPTV subscribers. We begin by describing the entities in our system together with their goals and knowledge.

[2] 95/46/EC requires subscribers' informed consent and proper data protection techniques such as encryption and anonymization before disclosure and/or processing.

4.1 Entities

Following the terminology introduced in EC Directive 95/46 [20], we assume that there are four entities in our system, namely data controller (IPTV service provider), data subjects (subscribers), data processor (cloud server) and recipient (advertiser).

Data controller (IPTV) Data controller is defined as any natural or legal person which determines the purposes and means of the processing of personal data. In our setting, the data controller is the IPTV, which is an entity that provides content to subscribers. It collects information about subscribers' demographics and watching habits, which are stored in a database. Each individualized entry, called subscriber profile, is considered private information. Statistics of subscriber profiles in aggregate form can be released as long as individual entries cannot be recovered from the released information. When it is considered as its legitimate interest, the data controller is willing to utilize any type of information which does not violate the privacy of individual subscribers for secondary processing. Furthermore, the data controller aims to reduce management costs by outsourcing database storage, backup and management issues to a third party. Outsourcing can lead to data protection risks, and therefore, the data controller should conform to privacy regulations [20].

Data subject (subscriber) Data subject is an identifiable person who can directly or indirectly be identified, in particular by reference to factors specific to his physical, economic, cultural or social identity [20]. In our setting, the data subjects are the subscribers, which are the customers of the IPTV (i.e., the data controller). Privacy-preserving data mining operations are applied on the encrypted subscriber profiles, and only the most related subscribers receive an advertisement from the advertiser. The privacy of the subscriber profiles is protected throughout the scheme utilizing several cryptographic techniques.

Data processor (cloud server) Data processor is defined as any entity which processes personal data on behalf of the controller [20]. In our setting, the cloud server (CS for short) is the data processor which is a professional entity that offers computing and storage services to any party according to specifications provided by the said party. The CS does not deviate from the provided specification, but curious to infer any information from the use of its services. This behavior of the server can be exercised on legal basis by making an agreement with the data controller. We assume that it is against the business interest of the CS to collude with any entity against other entities to violate the privacy of any individual since it is legally responsible by the 95/46/EC directive for any such violation. As such, the CS is what is known as "honest-but-curious" entity.

Recipient (advertiser) Recipient is any person or authority to whom data are disclosed. The recipients are the advertisers

which are entities that analyze demographics and watching habits of the subscribers and map the advertisements accordingly. Advertisements are generated based on a private advertiser strategy (e.g., data mining rules/techniques) that requires information about the target subscribers as input. Therefore, the advertiser is willing to utilize any database that contains subscriber profiles, such as a database generated by the IPTV. Since mining rules can reveal information about the advertiser's strategy, the advertiser wants to keep those rules secret.

4.2 Framework of interaction

The general framework is illustrated in Fig. 1. The IPTV provides personalized program contents to the subscribers and collects information about them. In order to reduce the management costs, the IPTV outsources its database management to a CS. Since CS is managed by third parties, the database can only be stored in a form which does not violate the privacy of individual subscribers, for example after anonymization and encryption. Furthermore, the CS should not be able to perform useful mining queries without the assistance and the permission of the IPTV. Any prior IPTV assistance should be useful only to the designated entities. Moreover, the result of an advertiser query should not uniquely identify a subscriber. Hence, the actual attributes of the matching results should return only if the number of matching subscribers is sufficiently high. Otherwise, only aggregated information should be returned.

The advertiser obtains access to the database stored on the CS and perform data mining analysis using assistance from the IPTV. In the proposed solution, the database is stored in encrypted form on a CS, and therefore, the advertiser needs certain trapdoor information generated by the IPTV in advance. This trapdoor information should not be transferable. In addition, since the advertiser's mining rules are considered trade secrets, the CS should not be able to infer any information about advertiser's queries. After learning the data mining results from the CS, advertiser sends the advertisement together with the corresponding list of subscribers to the IPTV.

The CS in advance obtains protocol specification from the IPTV, and according to this specification, it responds to the requests of the authorized advertisers. Throughout these interactions, the CS does not collude with any entity to violate another entity's secret or private information. Furthermore, it does not deviate from the provided specifications.

5 Preliminaries

Next, we consider some of the well-known approaches used in data mining community for filtering, anonymizing and obfuscating the data.

5.1 Data filtering techniques

The data mining algorithms for filtering data can be useful in matching prospective subscribers with advertisements. Intuitively, these techniques filter out subscribers, who would not be interested in an advertisement or whose interests would not exceed a certain threshold. Other methods or constraints can also be applied for selecting the set of advertisement receivers, as well. This may depend on the time of the day or

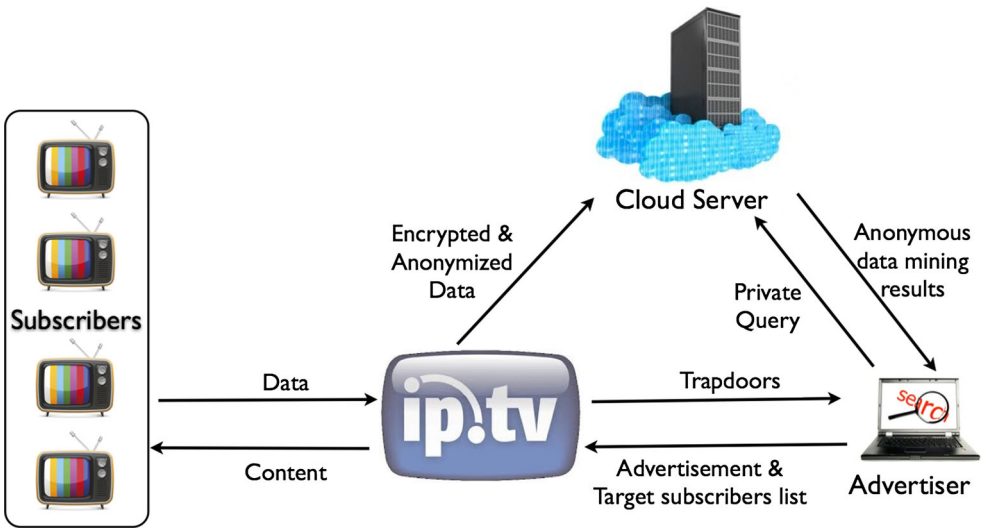

Fig. 1 General framework of directed advertising service

season (e.g., holiday, summer). These are natural constraints that can be applied in a straightforward manner that is in line with sector-specific practices. Privacy implications in these methods are not identified and thus not addressed in this paper.

5.1.1 Content-based filtering

Each advertisement can be described by some attributes or characteristics of the product (or service), which are the subject of the advertisement. The advertiser can use a similarity metric between the advertisement attributes and attributes of the IPTV content to predict subscribers' potential interest in a specific advertisement. The advertiser uses a public database to obtain the attributes of the IPTV content (e.g., from a movie database for the movies offered as video-on-demand service by the IPTV). A similarity metric between the attributes of the IPTV content and the advertisement attributes is determined by the advertiser, specifics of which are beyond the scope of this work.

In content-based filtering, while utilizing the similarity metric, the advertiser calculates a prediction for subscribers' potential interest in a given advertisement. The similarity function $\mathcal{S}(x, y)$ produces a real number in the range [0, 1], where similarity between x and y increases as $\mathcal{S}(x, y)$ gets higher. Consider an advertisement c, which has the assumed nonzero similarities to the IPTV contents $(m_1, m_2, \ldots, m_\sigma)$. Then the prediction for the interest of a subscriber s on the advertisement c can be calculated as [21]:

$$P_{c,s} = \frac{\sum_{i=1}^{\sigma} \left(\mathcal{R}_{s,m_i} \cdot \mathcal{S}(m_i, c)\right)}{\sum_{i=1}^{\sigma} \mathcal{S}(m_i, c)}, \tag{1}$$

where $\mathcal{R}_{s,m_i}$ is the interest score of the subscriber s on the IPTV content m_i. Since the interest of an individual subscriber on a specific IPTV content is not available for privacy protection, the prediction for a subscriber cannot be computed on individual basis. However, we can instead determine the required interest levels on certain IPTV contents, which show sufficiently high similarities to the advertisement c. Then, the subscribers that show sufficiently high interest on these IPTV contents can be targeted for the advertisement c. To this end, the advertiser defines a rule that determines the query content.

Example 1 Suppose that an advertisement for a perfume brand which is advertised with a famous actress will be sent to the subscribers who watch all the movies of that actress. For a high prediction value for the interest of prospected subscribers, the advertiser computes the desired level of interest on the movies by the actress. The rule is then, "subscribers who watch all the movies of that actress." A single query that contains the trapdoors for all the movies by that actress will determine the subscribers who will be the target of the advertisement.

Example 2 Following Example 1, if the advertiser defines the rule as "subscribers who watch and like at least four movies of the given actress," then a separate query for each of her movies will be sent to the cloud server along with the anonymized rule which is "find out the profiles who match at least four of the queries with a certain relevancy score." Only the subscribers, whose relevancy scores for at least four of such movies are higher than a threshold, will be the targets of this advertisement.

5.1.2 Collaborative filtering

Collaborative filtering [22] can be used to recommend a new movie or content in IPTV services such as video-on-demand, and thus, it is a type of advertising, where the IPTV itself can be the advertiser. It is based on the preferences of other subscribers with similar watching preferences.

Suppose that the advertiser has a catalog of $\mathcal{K}$ movies and he can query the cloud server to find the subscribers who watched and rated the movies in its catalog. The advertiser's aim is to find potential subscribers to recommend a movie m. To find such a subscriber s, who has not watched m yet and is likely to enjoy it, the advertiser establishes a similarity relation with other subscribers over the movies commonly watched and rated. Pearson correlation [23], ρ, is used to find such a similarity between two subscribers s and u as follows:

$$\rho(s, u) = \frac{\sum_{j \in \mathcal{J}} (\mathcal{R}_{s,j} - \bar{\mathcal{R}}_s) \cdot (\mathcal{R}_{u,j} - \bar{\mathcal{R}}_u)}{\sqrt{\sum_{j \in \mathcal{J}} (\mathcal{R}_{s,j} - \bar{\mathcal{R}}_s)^2 \cdot \sum_{j \in \mathcal{J}} (\mathcal{R}_{u,j} - \bar{\mathcal{R}}_u)^2}}, \tag{2}$$

where $\mathcal{J}$ is the set of movies which are rated by both subscribers, $\mathcal{R}_{l,j}$ is the rating of subscriber l on a movie j, $\bar{\mathcal{R}}_l$ is the average rating of a subscriber l for the movies in $\mathcal{J}$, and $l \in \{s, u\}$. The result will be a real number in the range $[-1, 1]$ indicating negative correlation and perfect match, respectively. The Eq. (3) is used to predict the rating of subscriber s on a movie $m \notin \mathcal{J}$

$$P(s, m) = \bar{\mathcal{R}}_s + \frac{\sum_{u_i \in U} \rho(s, u_i) \cdot (\mathcal{R}_{u_i,m} - \bar{\mathcal{R}}_{u_i})}{\sum_{u_i \in U} |\rho(s, u_i)|}, \tag{3}$$

where U is the set of subscribers who rate movie m, $s \notin U$ and $\mathcal{R}_{l,j}$ and $\bar{\mathcal{R}}_l$ are as defined above.

Example 3 Suppose that advertiser wants to find out subscribers to recommend the animated movie "How to Train Your Dragon," which is similar to "Madagascar 3" and "Kung

Fu Panda 2."[3] First, advertiser sends three query words, each of which contains the trapdoor for the corresponding movie. Inspecting the matching profiles and their relevance scores, the cloud server calculates the prediction for the rating for "How to Train Your Dragon" for those subscribers who have not watched it, but have watched the other two movies. Alternatively but less accurately, the advertiser forms two queries, one of which contains the trapdoors for all three movies and the other contains only those for "Madagascar 3" and "Kung Fu Panda 2." The cloud server finds out subscribers who have not watched "How to Train Your Dragon," but given high scores to the other two if there are sufficient numbers of other subscribers who rated all three movies with high scores.

5.2 Data anonymization

We use the Cornell Anonymization Toolkit [24], designed by Cornell Database Group at Cornell University for anonymizing published data set to restrict the disclosure of records against different types of attacks.

In our method, described in Sect. 7.2, we assume that the IPTV anonymizes the data before publishing it in the cloud server. To observe how much information can be leaked after anonymization of the data, we apply the Cornell Anonymization Toolkit (CAT) on the real data of the MovieLens data set [25].

CAT anonymizes data with techniques which generalize nonsensitive attribute values into value ranges specified by a user. This toolkit provides an ability to estimate the disclosure risk of each record in anonymized table based on user's assumptions about an adversary's background knowledge. In our experiments, we utilize the l-diversity method provided by the CAT.

Intuitively, if an anonymized block of data is l-diverse, there are at least l well-represented values for each sensitive attribute. An adversary cannot infer the sensitive information from an l-diverse data since there are at least l possible choices.

The anonymization method provides the anonymity of the subscribers against the advertisers. In the next section, we propose a method that obfuscates the data set in order to hide the sensitive content of the queries from the cloud server.

5.3 Obfuscation

A cloud server might have some background knowledge about the statistics of the IPTV's data set. In this case, the cloud server can learn or estimate some of the keywords in the queries if the results correlate with the statistics. For instance, in the MovieLens data set [25], about 72 % of subscribers are male which is the only attribute that occurs in such a high frequency. When the result of a query is such that 72 % of the data set is matched, then the cloud server can infer that the query contains "Male" keyword. In order to break the correlation between the original known statistics and the statistics of the query results, we propose obfuscating the data by insertion of fake profiles. The IPTV analyzes the statistics of the data set and generates a number of fake profiles accordingly. Since the fake profiles that are inserted to the data set are indistinguishable from the genuine profiles, the cloud server cannot correlate the data set statistics with the query results. However, the statistical results that the advertiser receives do not include the fake profiles. The method for the generation of fake profiles is discussed in Sect. 7.5.

6 Privacy requirements

A privacy-preserving advertising scheme should satisfy certain privacy requirements in order to protect the privacy of the advertisers' queries and the subscribers' data. Intuitively, the searchable encrypted index file of the subscribers' anonymized data should not leak sensitive information about the subscribers while the private queries of the advertisers should not leak sensitive information such as the queried terms. We provide formal privacy definitions for a privacy-preserving targeted advertising scheme. The proofs that indicate our proposed scheme is privacy-preserving are given in Sect. 10.

Definition 1 (*Trapdoor nontransferability*) A privacy-preserving targeted advertising scheme provides trapdoor nontransferability, if an advertiser cannot apply search using a trapdoor that is purchased by another advertiser, without revealing either the query terms or the private key of the advertiser that owns the trapdoor.

Definition 2 (*Access control*) A privacy-preserving targeted advertising scheme provides access control if any entity that is not registered to the IPTV cannot impersonate a registered advertiser.

Definition 3 (*Statistical privacy*) Let an adversary have the information of occurrence frequencies of each keyword in the data set. A privacy-preserving targeted advertising scheme provides statistical privacy, if it is hard to determine the keywords encoded in a query by such an adversary with this a-priory knowledge.

The following security definitions on searchable encryption, provided bellow for completeness, are due to Curtmola et al. [26].

[3] Similarity among movies can be determined by the advertiser in any arbitrary way by inspecting a public movie database such as IMDB. For instance, the movies by the same studio can be considered highly similar as in the example above.

The search pattern is the frequency of the queries searched. It can be found by checking the equality between two queries.

Definition 4 (*Search pattern* (S_p)) Let $\{Q_1, \ldots, Q_n\}$ be a list of queries. Search pattern S_p is an $n \times n$ binary matrix, where

$$S_p(i, j) = \begin{cases} 1, & \text{if } Q_i = Q_j \\ 0, & \text{otherwise.} \end{cases}$$

Intuitively, any deterministic query generation method reveals search pattern.

Definition 5 (*Access pattern* (A_p)) is the collection of data identifiers that contains search results of an advertiser query.

Let L_i be the keyword list of Q_i and $R(L_i)$ be the collection of the identifiers of data elements that match with keyword list L_i, then $A_p(Q_i) = R(L_i)$.

Definition 6 (*History* (H_n)) Let $\mathcal{D}$ be the data set of subscribers' information and $\mathcal{Q} = \{Q_1, \ldots, Q_n\}$ be a collection of n consecutive queries. The n-query history is defined as $H_n(\mathcal{D}, \mathcal{Q})$.

Definition 7 (*Trace* $(\gamma(H_n))$) Let $C = \{C_1, \ldots, C_l\}$ be the set of encrypted subscriber data, $id(i)$ be the identifier of the ith subscriber and $|C_i|$ be the size of C_i. Furthermore, let $||Q_j||$ be the number of keywords in query Q_j, $\mathtt{sig}(Q_j)$ be the digital signature of query Q_j, $|\mathtt{sig}(Q)|$ be the size (i.e., modulus) of the signature method, κ be the number of attributes in a subscriber data, $\mathcal{I}$ be the searchable index and $||\mathcal{I}||$ be the number of all elements, fake and genuine, in $\mathcal{I}$.

The trace of H_n is defined as

$$\gamma(H_n) = \{(id(1), ..., id(l)), (|C_1|, ..., |C_l|), |\mathtt{sig}(\mathcal{Q})|, \\ (||Q_1||, \ldots, ||Q_n||), \kappa, ||\mathcal{I}||, S_p(H_n), A_p(H_n)\}.$$

We allow to leak the trace to an adversary (e.g., the IPTV in this case) and guarantee no other information is leaked.

Definition 8 (*View* $(v(H_n))$) is the information that is accessible to an adversary from a history H_n. Let $\mathtt{sig}(\mathcal{Q})$ be the list of the signatures of queries in $\mathcal{Q}$ and, $id(i)$ and $\mathcal{Q}$ are as defined above. The view of H_n is defined as: $v(H_n) = \{(id(1), \ldots, id(l)), C, \mathcal{I}, \mathcal{Q}, \mathtt{sig}(\mathcal{Q})\}$.

Definition 9 (*Semantic security*) A cryptosystem is semantically secure, if for all probabilistic polynomial time algorithms (PPTA) there exists a simulator $\mathcal{S}$ such that given the trace of a history H_n, $\mathcal{S}$ can simulate $v(H_n)$ with probability $1 - \epsilon$, where ϵ is a negligible probability.

Intuitively, all the information accessible to an adversary (i.e., view of the n-query history) can be constructed from the trace that is allowed to leak.

Definition 10 (*Privacy-preserving targeted advertising scheme*) An advertising scheme is called privacy-preserving under the semantic security model, if it provides: semantic security, statistical privacy, trapdoor nontransferability and access control.

7 Construction

We provide the details of the proposed method in this section.

7.1 Data model

The data model we use in the proposed scheme is a generic, keyword-based model. Any attribute or feature is represented by a keyword in the model. Specifically, for every subscriber, we create a profile that basically consists of keywords corresponding to his/her demographics and watching preferences. For example, attributes such as gender, age, occupation and the IPTV content being watched can be represented through specific keywords. The collection of profiles for all subscribers in the IPTV service forms the so-called searchable index file.

Since watching preferences should incorporate the information about how much a certain IPTV content is preferred by a subscriber, the index for watching preferences contains the ranking level for a particular content. For the ranked search, i.e., the relevancy of the profiles to the query, we use η rank levels such that the higher the rank level, the higher the relevancy (e.g., more frequently watched). The construction of η-ranked search is described in Sect. 7.2.

In order to test the accuracy of the proposed method, we used a publicly available real data set, MovieLens [25]. We provide the properties of the real data set as an example to show how a database can be represented in our data model.

Real Data. MovieLens data set [25] is a widely used public database which is collected by the GroupLens Research. We use "Movielens 1M" data set, which consists of three files: users, ratings and movies, where there are 1,000,209 ratings (in 1–5 scale) from 6040 users on 3900 movies. The *users* file contains demographic information about users/subscribers in the following form:

$$[UserID, Gender, Age, Occupation, ZIP\text{-}code].$$

The values for *Age* attribute are provided in range: {"Under 18", "18–24", "25–34", "35–44", "45–49", "50–55", "56+"}. The values for occupation can be one of the following 21 alternatives: academic/educator, artist, clerical/admin, college/grad student, customer service, doctor, executive/managerial, farmer, homemaker, student, lawyer, programmer, retired, sales/marketing, scientist, self-employed, tech-

nician/engineer, tradesman, unemployed, writer and other. The ZIP code is a five digit integer indicating user's ZIP code. The *rating* file contains information about ratings in the form: $[UserID, MovieID, Rate, Timestamp]$. Ratings are made on a 5-star scale, where values are integer numbers in the interval of [1,5] and correspond to our ranking levels indicating how much a content is preferred by an individual.

While we use the MovieLens data set to test the keyword probability distributions and accuracy rates of the method on a real data set, we also test the method on a larger-scale problem using a synthetic data set for testing the efficiency and scalability. The details of the synthetic data set and the test results are provided in Sect. 12.1.

The proposed data model is not rigid and does not impose any restriction on our solution. The demographic features and watching preferences can be modified and extended to real-world scenarios. For example, the entire household cannot be considered as a single individual (i.e., Female or Male). To provide more precise advertising services, it is better for the IPTV to know each member of household and follow their watching habits (e.g., when the IPTV observes that cartoons are being watched in the morning, it infers that there are children in the household). Additional categories such as family, children and adults can be defined for improving the precision of targeted advertisements. Identifying categories of households according to watching histories and following their watching habits can be done by the IPTV. Our data model can incorporate such a modification.

7.2 Index generation

The solution is inspired by the construction in [16]. The main idea is to represent each database record as a binary string referred to as *index entry*. Different from the method used in [16], we add ranking capability to the search method. To accommodate η-ranked search, each record is expanded to η *index entries*. Without loss of generality, there are five levels of rating in the $MovieLens$ data set assigned by the subscribers to the movies, all of which can be captured in our data model.

Subsequently, *index entry* for each database record is cloned η times, once for each rank level. In the case of the real data set, each clone of the subscriber record contains all the demographic information of the subscriber. However, according to the given rate, the names or genres of the movies are included only in the clones that exceed the minimum rate of that rank level. For example, while a low rank genre such as "horror" is only indexed in Rank 1 (e.g., rarely watched), a high rank genre like "comedy" can be indexed in multiple ranks such as, Rank 3, Rank 2 and Rank 1, according to the given ratings. Our intention for such a ranking system is that the advertisements targeting the subscribers that at least occasionally watch a genre should also be sent to the subscribers that frequently watch that genre.

While only index generation for rank 1 is described in the work, the indexes for other ranks are generated in an analogous way. The secure index is encoded using a keyed hash-based Message Authentication Code (HMAC) [27]. An HMAC function is a cryptographic hash function in combination with a secret cryptographic key. In the proposed model, it is not necessary to decrypt the secure index; hence, an efficient one-way encoding such as an HMAC function is more suitable compared to more costly encryption operations.

The IPTV selects an HMAC key, which is used to generate the secure index. Note that the same key is used for all the entries in the index, including the new entries added subsequently. However, for security reasons, the private key can be updated periodically, in which case the secure index is to be regenerated for the entire data set. For each profile, the maximum number of fields is limited by m. From this point onward, we name these fields as keywords and each database record as a profile. A profile may have less than m keywords.

Suppose a given profile at a given rank contains keywords $\{w_1, \ldots, w_n\}$, where $n \leq m$. The IPTV generates the corresponding rank index (binary string) by computing the l-bit HMAC h_i of each keyword w_i (HMAC: $\{0,1\}^* \rightarrow \{0,1\}^l$). Let

$$h_i = h_i^{r-1}, \ldots, h_i^1, h_i^0 \tag{4}$$

be the base 2^d representation of h_i $(l = rd)$. Using the h_i values of a keyword w_i, the IPTV applies a reduction operation and computes a binary string (keyword trapdoor) T_i as,

$$T_i = (T_i^{r-1}, \ldots, T_i^1, T_i^0), \tag{5}$$

where

$$T_i^y = \begin{cases} 0, & \text{if } h_i^y = 0 \\ 1, & \text{otherwise.} \end{cases} \tag{6}$$

The index entry for the jth profile at a given rank is

$$I_j = \odot_{i=1}^{n} T_i, \tag{7}$$

where $\odot$ denotes bitwise product.

The generated records have the form

$$\left(id(j), I_j^{rank\_1}, \ldots, I_l^{rank\_\eta}\right),$$

where $I_j^{rank\_k}$ is the index of the jth subscriber's profile at rank k and $id(j)$ is an anonymized pseudonym for the jth

subscriber. The searchable index $\mathcal{I}$ is the set that contains the index entry records of all the profiles. The IPTV sends the searchable index $\mathcal{I}$ to the CS.

The proposed system is suitable for updates in the data set. The addition of new subscribers to the system can be handled without changing the existing entries. Each subscriber's information is stored in a separate entry of the secure index, and it is independent from the other entries; hence, new subscribers can trivially be added or existing subscribers can be removed from the secure index without changing the data of other subscribers. Similarly, the proposed scheme does not use a predefined keyword dictionary set, and hence, any new keyword can trivially be added and used in the system by just applying the HMAC function and bitwise product operation as shown in Eq. (7).

In some settings, some subscribers may prefer to receive only few advertisements that come from a specific, small set of advertisers. IPTV may sell different subscription options such as "*premium subscriber*" and "*classical subscriber*," where *premium subscribers* receive advertisements from a restricted group of advertisers, while *classical subscribers* receive from a broader range. A trivial solution for this problem is to generate a different index for each subscriber set. Each index will be generated using only the attributes of the subscribers in the corresponding set, and advertisers can only apply search in the sets that corresponds to their access rights. This setting may require indexing some subscribers in multiple indexes which creates additional computation and storage costs as well as possible privacy implications. We did not consider the problem of multiple subscriber sets in this work and leave privacy and efficiency issues of this problem as a future direction.

7.3 Query index generation and oblivious search on the database

The IPTV generates a trapdoor per each keyword, and an advertiser obtains the trapdoors for any set of keywords in the database entries from the IPTV. Later, the advertiser can generate a query using any subset of the purchased trapdoors and submit this query to the CS.

In the index generation scheme, multi-keyword (i.e., conjunctive keywords) queries can be efficiently constructed, resulting in an r-bit binary sequence independent of the number of keywords in the query. A multi-keyword query index of δ keywords, $\{w_1, \ldots, w_\delta\}$, is the bitwise product of the corresponding trapdoors $\{T_1, \ldots, T_\delta\}$:

$$I^q = \odot_{i=1}^{\delta} T_i = q_{r-1} \ldots q_0.$$

A query from the advertiser to the CS is such an r-bit binary sequence known as query index, and search is done only by r-bit comparisons. The result is a list of subscriber identifiers ($id(i)$) such that the corresponding index value of a subscriber matches with the query. An index entry, $I_k = j_{r-1} \ldots j_0$ in the list, matches with a query index, $I^q = q_{r-1} \ldots q_0$ if the index entry (I_k) has 0 for all the bits, for which the query index has 0 as shown in Eq. (8).

$$result(I_k, I^q) = \begin{cases} \text{match} & \text{if } \forall i, q_i = 0 \Rightarrow j_i = 0, \\ \text{not match} & \text{otherwise.} \end{cases} \tag{8}$$

In the case of disjunctive queries, the trapdoors for query keywords are sent separately; therefore, a separate result is generated for each keyword in the query. The corresponding results need to be combined with other search results by appropriate operations such as union and subtraction.

Results of the matching records can be returned in a ranked manner. The CS returns separate anonymized id lists for each rank level (e.g., seldom, average and frequent ranks).

In the case, where a stronger privacy is required, for example due to existence of background information, only statistical data are returned as explained in the subsequent sections.

In the proposed method, the CS is oblivious to the actual outsourced data since the outsourced information is limited to anonymized pseudo-identifiers of subscribers and the corresponding secure searchable index entries per each rank, as

$$\left(id(i), I_i^{rank\_1}, \ldots, I_i^{rank\_\eta}\right).$$

The queries are also in the same structure with the secure index, and therefore, the CS cannot learn the queried keywords from the r-bit binary query without knowing the secret key used in the cryptographic hash computation. Hence, the CS is oblivious to the queries. Finally, the CS returns the list of pseudo-identifiers corresponding to the matching subscribers. Although this leaks both the search and access patterns as defined in Sect. 6, the CS is nevertheless oblivious to the responses since neither the identities of the actual subscribers nor their traits and demographics are revealed.

7.4 Trapdoor nontransferability

The trapdoor nontransferability protocol is designed to prevent advertisers from sharing their trapdoors with other parties (or using other parties' trapdoors in their queries). This protocol stipulates that the advertiser submits a proof of ownership for the trapdoors in its queries. Initially the advertiser creates a single secure index T, for all the trapdoors he purchased as described in Algorithm 1. The

IPTV creates a signature for this secure index and sends it to the advertiser. Whenever the advertiser wants to make a query, the combined index of the purchased trapdoors and its signature are both sent to the CS together with the query. The CS first verifies the signature and tests whether the query is generated from a subset of the trapdoors in T. The search is applied only if both tests are verified.

Note that the signature is generated only once for each advertiser after obtaining trapdoors. However, the CS verifies the signature before each query.

Algorithm 1 Trapdoor Nontransferability Protocol

1: Keys of Participants:
2: PU_X- public key of the party X
3: PR_X- private key of the party X
4:
5: IPTV:
Require: trapdoors $(T_1, T_2, \ldots, T_t)$ requested by an advertiser, corresponding to the keywords $(w_1, w_2, \ldots, w_t)$
6: $T = \odot_{i=1}^{t} T_i$
7: $= \{j_{r-1} \ldots j_0\}$
8: {signature of the purchased trapdoors}
9: $S = SIGN_{PR_{IPTV}}(T, PU_{ADV})$
10: send (S, T) to advertiser
11:
12: Advertiser:
Require: keyword list for query $\{w_1, w_2, \ldots, w_\delta\}$
13: **if** S is valid **then**
14: $I^q = \odot_{i=1}^{\delta} T_i$
15: send (S, T) and I^q to CS
16: **end if**
17:
18: CS:
19: **if** S is valid **then**
20: {this loop checks if $I^q \subseteq T$}
21: **for all** $i \in [0, \ldots, r-1]$ **do**
22: **if** $I_i^q = 0 \wedge j_i \neq 0$ **then**
23: {advertiser not authorized for querying I^q}
24: abort
25: **end if**
26: **end for**
27: Generate response R by performing I^q on the data set
28: Select a random k as symmetric key
29: send $ENC_{PU_{ADV}}(k)$ and $ENC_k(R)$ to advertiser
30: **end if**
31:
32: Advertiser:
33: $k = DEC_{PR_{ADV}}(ENC_{PU_{ADV}}(k))$
34: $R = DEC_k(ENC_k(R))$

The trapdoor nontransferability property also allows advertiser revocation. If the contract of an advertiser is expired or for any other reason needs to be revoked, the public key of that advertiser is removed from the CS side. Therefore, the old signature cannot be verified and the corresponding query will not be processed by the CS.

7.5 Fake profile generation

As mentioned in Sect. 5.3, the IPTV can obfuscate the data by adding some fake profiles to the database. Then, both genuine and fake profiles are sent to the CS after encryption. This obfuscation is used to prevent the CS from learning keywords in a query from the statistics of the database. A fake profile is added, if the frequency of an attribute a in the profiles is not within the range $[t_1, t_2]$, where t_1 and t_2 are predefined thresholds. The frequency of an attribute a is calculated as:

$$fq_a = \frac{\#\text{ of profiles containing } a}{\text{total number of profiles (fake + genuine)}}.$$

Whenever the CS searches for a query over encrypted data, the fake profiles are also included in the query results, but only genuine profiles are revealed to the advertisers. The algorithm used to create fake profiles is illustrated in Algorithm 2.

Attributes with smaller frequency have greater probability to appear in the fake profiles. At each iteration, a block of θ fake profiles is generated in order to reduce the number of while condition checks and hence increase the efficiency. As the attributes in the fake profiles are chosen from the ones with frequencies smaller than t_1, only the frequencies of attributes that are smaller than t_1 increase, while all the others decrease. There may be a case that the frequency of an attribute, which is initially greater than t_1, decreases below the threshold after the addition of fake profiles. Then, this attribute becomes a candidate for the fake profile addition in the next iteration. A threshold can be used for the maximum number of iterations to guarantee termination of the algorithm.

Algorithm 2 Fake profile generation

Require: thresholds t_1,t_2
A: the set of all possible values for all attributes
1: **for all** $a_i \in A$ **do**
2: IPTV calculates fq_{a_i}
3: **end for**
4: **while** $\exists a_i \in A$ such that $fq_{a_i} < t_1$ OR $fq_{a_i} > t_2$ **do**
5: **for all** a_i such that $fq_{a_i} < t_1$ **do**
6: calculate $P'_{a_i} = t_1 - fq_{a_i}$
7: **end for**
8: **for** $i = 1 \rightarrow \theta$ **do**
9: Create a fake profile such that attributes that has larger P'_{a_i} have higher chance to be appeared in the profile. Each profile contains 5 levels such that number of keywords in each level of a profile are selected with respect to the distribution in the original data set.
10: $i \leftarrow i + 1$
11: **end for**
12: $\forall a_i \in A$, $recalculate\ fq_{a_i}$
13: **end while**

The pseudocode of the overall protocol is illustrated in Algorithm 3.

Algorithm 3 Privacy-Preserving Advertising Scheme

IPTV: (offline stage)
Require: raw data of subscribers (D)
$D' = anonymize(D)$ {cf. Sect. 5.2}
$D'' = FakeProfGen(D')$ {cf. Sect. 7.5}
$\mathcal{I} = GenIndex(D'')$ {cf. Sect. 7.2}
send $\mathcal{I}$ to CS

Advertiser:
Require: keyword list for the requested trapdoors $(w_1, w_2, \ldots, w_t)$
send keyword list to IPTV and request corresponding trapdoors

IPTV:
Require: keyword list $(w_1, w_2, \ldots, w_t)$ from advertiser
generate trapdoors $(T_1, T_2, \ldots, T_t)$ requested by the advertiser, corresponding to the keywords $(w_1, w_2, \ldots, w_t)$
{trapdoor generation is given in Sect. 7.2}
create S and T { cf. Algorithm 1}
send $((T_1, T_2, \ldots, T_t), S, T)$ to advertiser

Advertiser:
Require: $((T_1, T_2, \ldots, T_t), S, T)$ and keyword list for query $\{w_1, w_2, \ldots, w_\delta\}$
validate S and create query I^q {cf. Algorithm 1}
send (S, T, I^q) to CS

CS:
validate S and check if $I^q \subseteq T$ {cf. Algorithm 1}
for all $I_i \in \mathcal{I}$ **do**
  calculate $result(I_i, I^q)$ {cf. Sect. 7.3}
end for
Generate response R by selecting the positive results
Select a random k as symmetric key
send $ENC_{PU_{ADV}}(k)$ and $ENC_k(R)$ to advertiser

Advertiser:
$k = DEC_{PR_{ADV}}(ENC_{PU_{ADV}}(k))$
$R = DEC_k(ENC_k(R))$

8 A stronger protection scheme in the presence of background knowledge

In this section, we explain a key technique to provide a stronger protection for the case, where there is some background knowledge on subscribers or the statistical model of the data set.

To protect the privacy of subscribers' data, the IPTV has to anonymize the data before uploading it to the CS. If the data are not anonymized properly, an advertiser who obtains all the keywords from the IPTV can easily obtain profiles of the subscribers by aggregating the result of queries. Practical anonymization techniques do not eliminate the risks completely. An adversary who has access to the anonymized database and some auxiliary information about a subscriber can identify a record in the anonymized database and learn the complete viewing history [28]. The advertiser can learn the pseudonym of a target subscriber and compile his complete viewing history by asking related queries. Therefore, in such cases it is safer to provide only some statistical knowledge about the result of a query, instead of returning the pseudonyms of subscribers. Using only statistical knowledge, an advertiser is not able to track a specific subscriber and learn additional information.

For reducing privacy risks, in addition to the index of a profile, some statistical information (e.g., age, gender, occupation, etc.) is also encrypted using a cryptographic algorithm with homomorphic properties [29] before uploading the data to the CS. Whenever the CS receives a query, utilizing the homomorphic properties, encrypted sum of statistical information is calculated by the CS. Then the encrypted statistics are returned to the advertiser, which are, in fact, aggregated data. As an example, statistics of a query result can include female and male distribution of the matching profiles. In the MovieLens data set, we consider genuine/fake, female/male, age and occupation attributes as the statistical information. Moreover, the security can further be increased by separating the statistics of more sensitive features from the others. While the aggregated results of sensitive data are returned only if the number of matching profiles is sufficiently high, the aggregated results of the features, which are not considered to be sensitive, can always be returned independent from the number of matching profiles. As an example, we can separate occupation, which is *arbitrarily* assumed to be more sensitive than gender and age. Statistical information of a subscriber i is represented as:

$$S_{i1} = (P_{i1}||P_{i2}||\ldots||P_{in_1}) \tag{9}$$

$$S_{i2} = (\bar{P}_{i1}||\bar{P}_{i2}||\ldots||\bar{P}_{in_2}), \tag{10}$$

where each of P_{ik} and $\bar{P}_{ik}$ parts is a t-bit integer, representing statistical information of a subscriber i and n_1, n_2 are the number of fields in the corresponding statistical information.

For example, let in Eq. (9), there be seven fields (i.e., $n_1 = 7$). P_{i1} shows whether a subscriber is genuine or fake, P_{i2} and P_{i3} specify if subscriber is female or male. The four anonymized ranges of ages, namely (0-24), (25-44), (45-64) and (65+) are represented in P_{i4} to P_{i7}, respectively. In addition, let there be 21 different occupations, which are more sensitive and hence represented in S_{i2} by $\bar{P}_{i1} \ldots \bar{P}_{i21}$ as in Eq. (10). Since each subscriber is assumed to have a single occupation, only the $\bar{P}_{ik}$ field, corresponding to the occupation of the ith subscriber, is set to one, while the rest of the fields are set to zero.

The IPTV encrypts S_{i1} and S_{i2} separately for all subscribers, using the (homomorphic) Paillier encryption [30] as shown in Eq. (11), where modulus n and the base $g \in \mathbb{Z}^*_{n^2}$ are the public keys and $r \in_R \mathbb{Z}^*_n$ is a random number that will be different at each encryption.

$$E(S_{i1}) = g^{S_{i1}} \cdot r^n \mod n^2$$
$$E(S_{i2}) = g^{S_{i2}} \cdot r^n \mod n^2 \quad (11)$$

Then, the IPTV sends the encrypted information along with the indexes of the profiles to the CS.

The CS calculates the aggregated data for the set of matching subscribers. Let $\mathcal{R}$ be the number of subscriber profiles matched with the query, the encrypted aggregated results S_j are calculated by multiplying the encrypted value S_{ij} of each subscriber i. Note that due the homomorphic properties of the Paillier encryption method, the following equations on the encrypted data holds.

$$\begin{aligned} S_1 &= \prod_{i=1}^{\mathcal{R}} E(S_{i1}) = E\left(\sum_{i=1}^{\mathcal{R}} S_{i1}\right) \\ &= E\left(\sum_{i=1}^{\mathcal{R}} P_{i1} || \sum_{i=1}^{\mathcal{R}} P_{i2} || \ldots || \sum_{i=1}^{\mathcal{R}} P_{in_1}\right) \\ S_2 &= \prod_{i=1}^{\mathcal{R}} E(S_{i2}) = E\left(\sum_{i=1}^{\mathcal{R}} S_{i2}\right) \\ &= E\left(\sum_{i=1}^{\mathcal{R}} \bar{P}_{i1} || \sum_{i=1}^{\mathcal{R}} \bar{P}_{i2} || \ldots || \sum_{i=1}^{\mathcal{R}} \bar{P}_{in_2}\right) \end{aligned} \quad (12)$$

Here S_{ij} is the statistics of a matching subscriber i and $j \in \{1, 2\}$ is the representative of each statistic as described in Eqs. (9) and (10). Although each P_{ik} and $\bar{P}_{ik}$ keeps a single bit information, t bits are reserved for each of them. Each S_j is aggregated according to the summation in Eq. (12); hence, the extra reserved space prevents any overflow from one part in S_j to the prior part during the aggregation.

The value of t can be set depending on the size of the data set. We assume that the number of matching subscribers will be less than 2^{16}; hence, in our experiments we set $t = 16$. However, the number of bits for each field can trivially be increased depending on the size of the data set. As a common message space for the Paillier encryption is 1024 bits, up to 50 bits per field can be used if required and this will have no effect on the time or space complexity of the system.

Advertiser can learn the statistics of the returned results by decrypting the encryption in Eq. (12). In order to decrypt the statistics, the IPTV needs to share the private key of the Paillier encryption with the advertisers. The IPTV sends this key to an authorized advertiser after encrypting with the public key of that advertiser that is also used in the trapdoor learning procedure given in Algorithm 1 and this is done only once per advertiser. Note that, due to the homomorphic properties of the Paillier cryptosystem, the encryption operations in Eq. (11) are only modular multiplications and thus can be performed very efficiently.

8.1 Privacy rules applied to returned statistics on sensitive attributes

Depending on sensitive attributes in the system, some privacy rules must be applied before returning any statistics about them. For instance, an advertiser (possibly with background knowledge) can arrange his query in such a way that only a unique subscriber matches. In this case, returning the sensitive attribute of the subscriber violates his privacy. We specify a threshold such that the values for sensitive attributes are returned to the advertiser only if the number of subscribers that match to a given query is greater than the threshold. In a sense, we apply a privacy rule similar to k-anonymity, in which response to every query should contain at least k profiles in order to return sensitive attributes.

The number of matching profiles alone, however, cannot determine the threshold since the matching profiles may not be sufficiently diverse in terms of sensitive attributes. For example, if all the subscribers in the set of matching profiles have the same occupation, returning statistics about the occupation attribute may violate the privacy of a subscriber with that occupation. The privacy rule, known as l-diversity, guarantees a sufficiently diverse set in terms of sensitive attributes. The number of subscribers matching to a query should be a relatively large integer $T > k$ to satisfy both k-anonymity and l-diversity. There is a trade-off between privacy and loss of information since higher threshold values mean better privacy, but less accuracy. However, our scheme does not really suffer from loss of accuracy since the advertisements are still sent to subscribers; only the advertiser does not learn sensitive information about the matching profiles except for the cardinality of the set of matching subscribers.

We experimentally determine an optimal value for the threshold T by making one million queries where two random keywords are assigned to each query from the MovieLens data set. Assuming that occupation and movie rating by subscribers (rank attribute) are sensitive attributes, we keep track of the number of matches and diversity in the sensitive attributes for each query. In Figs. 2 and 3, we present the histograms for diversity of occupation and rank for 1 million random queries with respect to the number of matches. The higher the number of matches, the higher the diversity for occupation and rank is. These results demonstrate that an optimal value for the threshold T is 13, which provides sufficient diversity for sensitive attributes.

Nevertheless, one needs to be careful about the rank attribute. The value for rank attribute is a number between 1 and 5. Statistics of ranks will be returned in the form Rank 1: N_1, Rank 2: $N_2 \ldots$, Rank 5: N_5, where N_i is the number of subscribers who give rate i to a queried movie (or set of movies). We can follow the l-diversity rule on rank results, where the value for l can be a number between 2 and 5. The

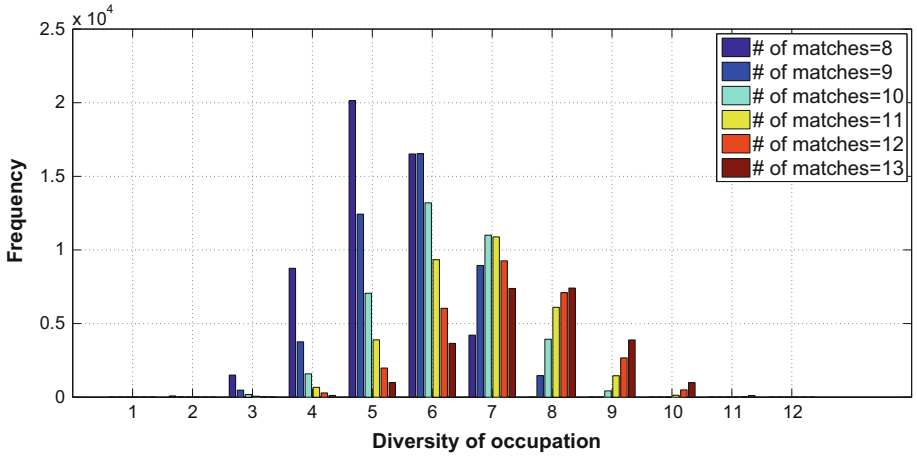

Fig. 2 Frequency of diversity of occupation with respect to the number of matches for 1 million random queries

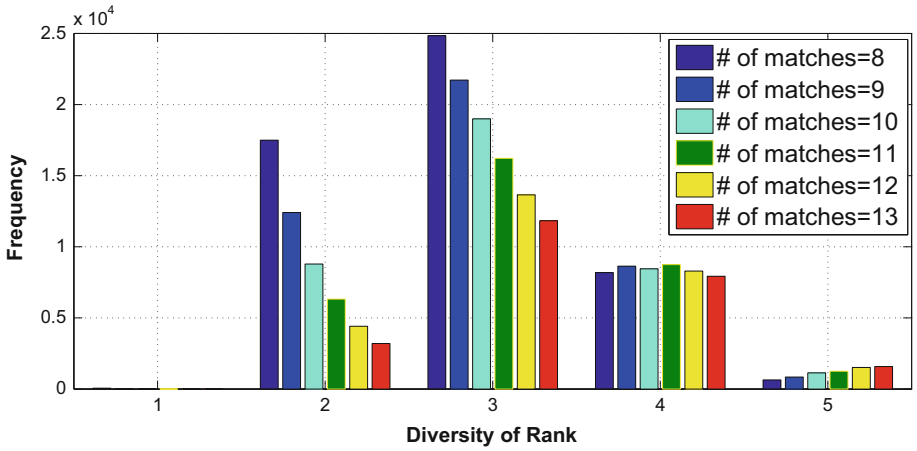

Fig. 3 Frequency of diversity of ranks with respect to the number of matches for 1 million random queries

statistics about ranks are only returned if rank diversity is greater than or equal to l.

From Figs. 2 and 3, setting the threshold $T = 13$ satisfies 2-diversity property (i.e., $l = 2$) for both occupation and rank diversity as none of the matches has an occupation or rank diversity less than l (i.e., 2).

9 Collaborative filtering over encrypted data

One of the most difficult data mining operations in our setting is collaborative filtering over encrypted data. In this section, the computation of collaborative filtering is explained in detail. As it is explained in Sect. 5.1.2, Eq. (3) is used to predict a rating of subscriber s on an IPTV content m (i.e., $P(s, m)$). The CS, which is supposed to calculate predictions, cannot identify the fake subscribers in the data set. Therefore, the predictions can be faulty if these fake subscribers are considered. Thus, we have to provide a technique that eliminates the effects of fake subscribers. The information of a subscriber being fake or genuine is first encrypted by the IPTV and provided to the CS in encrypted form. The CS calculates the prediction for a subscriber s (i.e., $P(s, m)$) as in Eq. (3). The predictions are calculated utilizing homomorphic operations. Each of the three parts of the equation is calculated separately as given in Eqs. (13), (14) and (15) and then combined as given in Eq. (16), where U is the set of subscribers that rate the movie m and $s \notin U$. Note that $\mathcal{R}_{u,m}$ is the rating of a subscriber u on content m, $\bar{\mathcal{R}}_u$ is the average rating of a subscriber u and $\rho(s, u)$ is the similarity between subscribers s and u.

$$E\left(\sum_{u_i \in U} (F_{u_i} \cdot |\rho(s, u_i) \cdot (\mathcal{R}_{u_i,m} - \bar{\mathcal{R}}_{u_i})|)\right) = \prod_{u_i \in U} E(F_{u_i})^{|\rho(s,u_i)\cdot(\mathcal{R}_{u_i,m} - \bar{\mathcal{R}}_{u_i})|} \tag{13}$$

$$E\left(\sum_{u_i \in U} (F_{u_i} \cdot |\rho(s, u_i)|)\right) = \prod_{u_i \in U} E(F_{u_i})^{|\rho(s,u_i)|} \tag{14}$$

$$E\left(\bar{R}_s \cdot \sum_{u_i \in U} |\rho(s, u_i)|\right) = \prod_{u_i \in U} E(F_s \cdot (\bar{R}_s \cdot |\rho(s, u_i)|)) = \prod_{u_i \in U} E(F_s)^{\bar{R}_s \cdot |\rho(s,u_i)|}. \tag{15}$$

Here F_i is fake/genuine indicator for a subscriber i (i.e., $F_{u_i} = 0$ if the profile is fake, 1 otherwise). The fake/genuine indicator allows the CS to eliminate the effect of fake subscribers without learning any information about fake subscribers. If $F_i = 0$, meaning that the subscriber with index i is fake, its ratings will have no effect on the aggregated result since they are multiplied with zero. It is important to emphasize that the encryption used for hiding F_i is probabilistic, so different encryptions of the same message with the same key produce different cipher texts. Hence, the CS cannot differentiate encrypted values of 0 from encrypted values of 1.

Recall that $P(s, m)$, which is the predicted rating of a subscriber s on content m, is calculated as in Sect. 5.1.2,

$$P(s,m) = \frac{\bar{\mathcal{R}}_s \cdot \sum_{u_i \in U} |\rho(s,u_i)|}{\sum_{u_i \in U} |\rho(s,u_i)|} + \frac{\sum_{u_i \in U} \rho(s,u_i) \cdot (|\mathcal{R}_{u_i,m} - \bar{\mathcal{R}}_{u_i}|)}{\sum_{u_i \in U} |\rho(s,u_i)|}.$$

Then the server calculates the encrypted prediction for the subscriber s as:

$$\begin{aligned} &E(P(s,m)) \\ &= \frac{E\left(\left(\bar{\mathcal{R}}_s \sum_{u_i \in U} |\rho(s,u_i)|\right) + \left(\sum_{u_i \in U} \rho(s,u_i)\left(|\mathcal{R}_{u_i,m} - \bar{\mathcal{R}}_{u_i}|\right)\right)\right)}{E\left(\sum_{u_i \in U} |\rho(s,u_i)|\right)} \\ &= \frac{E\left(\bar{\mathcal{R}}_s \sum_{u_i \in U} |\rho(s,u_i)|\right) E\left(\sum_{u_i \in U} \rho(s,u_i)\left(|\mathcal{R}_{u_i,m} - \bar{\mathcal{R}}_{u_i}|\right)\right)}{E\left(\sum_{u_i \in U} |\rho(s,u_i)|\right)} \\ &= \frac{E\,(num(P(s,m)))}{E\,(denom(P(s,m)))}. \end{aligned} \tag{16}$$

Encrypted nominator and denominator of $E(P(s, m))$ are calculated as follows:

$$E(num(P(s,m))) = \prod_{u_i \in U} E(F_s)^{\bar{\mathcal{R}}_s \cdot |\rho(s,u_i)|} \cdot E(F_{u_i})^{\rho(s,u_i)\cdot(\mathcal{R}_{u_i,m} - \bar{\mathcal{R}}_{u_i})} \tag{17}$$

$$E(denom(P(s,m))) = \prod_{u_i \in U} E(F_{u_i})^{|\rho(s,u_i)|}, \tag{18}$$

Encrypted nominator and denominator of $P(s, m)$ (i.e., Eqs. 17, 18) are returned by the CS separately. By decrypting those two equations, an advertiser can compute $P(s, m)$ for subscribers $s \in U_c$, where the set U_c includes subscribers $s \notin U$. Note that $s \in U_c$ are the pseudonyms for the subscribers who are the potential targets for the content m. The steps of the privacy-preserving collaborative filtering are given in Algorithm 4.

While calculating the predictions in collaborative filtering, the proposed method uses the same secure index structure as in the targeted advertising. The only extra information required is the encrypted fake/genuine identifier per subscriber (i.e., F_i). The values $\rho(s, u_i)$, $\bar{\mathcal{R}}_{u_i}$ and $\mathcal{R}_{u_i,m}$ are calculated by the advertiser, as follows: when the advertiser applies search with the trapdoor for a content m, the list of subscribers matching with m is received by the advertiser together with the corresponding ranks. By applying separate searches for different contents, the rates of subscribers on each content (i.e., $\mathcal{R}_{u_i,m}$) can be found, which are then used to find the average rating of each subscriber (i.e., $\bar{\mathcal{R}}_{u_i}$) and also the list of subscribers in the two sets U and U_c. The similarity between two subscribers $s \in U_c$ and $u \in U$ (i.e., $\rho(s, u)$) is then calculated using these ratings. The following example demonstrates the collaborative filtering process.

Algorithm 4 Privacy-Preserving Collaborative Filtering

for all keyword $j \in \mathcal{J}$ **do**
  Advertiser sends the query Q_j corresponding to keyword j to CS
end for
Advertiser sends the query Q_m corresponding to the keyword $m \notin \mathcal{J}$ to CS
for all keywords $j \in \mathcal{J}$ and $m \notin \mathcal{J}$ **do**
  CS performs oblivious search operations
end for
CS determines U and U_c from search results
CS calculates $\bar{\mathcal{R}}_u$
for all $s \in U_c$ **do**
  CS calculates $\bar{\mathcal{R}}_s$
  CS calculates $\rho(s, u)$
  CS calculates $E(num(P(s,m)))$ and $E(denom(P(s,m)))$ and sends them to advertiser
end for
for all $s \in U_c$ **do**
  Advertiser performs the decryption and calculates $P(s, m)$.
end for
Advertiser selects the genuine subscribers in U_c with high prediction values

Example 4 Suppose that there are four movies as: m_1 = "Brazil," m_2 = "Liar Liar," m_3 = "Barcelona" and m_4 = "Smoke." An advertiser would like to learn a prediction for subscribers who watched the first three movies, but not the forth. To do so, four queries, each containing one of the movies, are generated separately. The results of these queries are aggregated to generate a matrix which is partially shown in Table 1. According to this matrix, the CS can identify two sets of subscribers. The first set $U = \{308, 1015, 1019, ...\}$ contains the subscribers who rated all 4 movies, and the second set $U_c = \{151, 624, ...\}$ consists of the subscribers who rated the first three movies but not the forth, thus $U \cap U_c = \emptyset$. Then for each $s \in U_c$, the CS calculates Eqs. (17) and (18), where m is m_4="Smoke" and sends the results to the advertiser. If there exists a fake subscriber $u \in U$, then F_u will be 0, which eliminates its effect on the prediction. The advertiser can learn the prediction for each subscriber in U_c by dividing the corresponding results from the CS as shown in Eq. (16). The CS is not aware of which subscribers are fake or genuine, and the advertiser cannot gain any information

Table 1 Matrix of queries' results

ID	m_1	m_2	m_3	m_4
151	5	4	4	
308	5	3	3	5
624	2	5	5	
1015	5	2	3	4
1019	5	3	3	2
...	...	...	...	...

about the subscribers' ratings, other than the value for the prediction.

10 Privacy arguments

Here we argue that the proposed scheme addresses the privacy requirements in the setting defined in Sect. 6. We assume that the database schema (i.e., keywords) is known, but the adversary does not know the mapping between the keywords and trapdoors. We further assume that the statistical model of the database is initially known only by the data controller (IPTV). We analyze the security of the system against an adversary that can access all the communication between any two parties. Moreover, adversary may have some a-priory information on the data set such as the database size (i.e., number of profiles) and distribution statistics of separate attributes. The adversary also has access to the searchable index. In other words, the CS is the adversary, which is the strongest possible adversary in our setting.

Theorem 1 *The privacy-preserving targeted advertising scheme provides trapdoor nontransferability in accordance with Definition 1.*

Proof The trapdoor nontransferability protocol, which is explained in Sect. 7.4, prevents advertisers from utilizing unauthorized trapdoors in their queries. The index in the IPTV signature allows the CS to identify if the keywords in a given query indeed belong to the advertiser. This prevents an advertiser from querying the database with the keyword he/she did not own. Furthermore, the advertiser id in the IPTV signature allows the CS to identify which advertiser purchased the access to the database. The encryption that the CS uses prevents an advertiser A from selling keywords to another advertiser B. Indeed if advertiser B purchases keywords from the advertiser A rather than the IPTV, then either advertiser B has to reveal its query to advertiser A or advertiser A has to give its private key to advertiser B. In either case, one of the advertisers may be leaking some non-trivial information about their advertising strategies via their queries to a competitor. Thus advertisers have no incentives to collude against the IPTV.

Since the CS's response is encrypted using the advertiser's public key, only the designated advertiser learns the information transferred. Therefore, the proposed scheme provides trapdoor nontransferability. □

Theorem 2 *The privacy-preserving targeted advertising scheme provides access control in accordance with Definition 2.*

Proof Each query of advertisers is authenticated by the signature of the advertisers' private key. Assuming that the private keys of advertisers are not compromised, the probability of impersonating a registered advertiser is equal to the probability of breaking the underlying cryptographic signature scheme. At present, an RSA public key of at least 1024 bits would suffice to guarantee access control. □

Theorem 3 *The privacy-preserving targeted advertising scheme provides statistical privacy in accordance with Definition 3.*

Proof The addition of fake profiles, explained in Sect. 5.3, obfuscates the original distribution statistics of each attribute. After the addition of fake profiles, the occurrence of each attribute in the data set is guaranteed to be in the interval $[t_1, t_2]$, for two threshold parameters t_1 and t_2 such that $0 \leq t_1 < t_2 < 1$. Given that the occurrence rates of all attributes are obfuscated to be in the secure interval $[t_1, t_2]$, it is not possible for an adversary to map original distribution statistics of the data set to the distribution statistics of a query. Moreover, the generated fake profiles are indistinguishable from genuine profiles and adversary cannot learn the original distribution from the obfuscated distribution. Hence, the proposed scheme provides statistical privacy. □

We assume that the adversary has only access to distribution statistics of single attributes. In the case that the adversary has the knowledge of a joint occurrence (e.g., distribution statistics for age and gender such as "females that are younger than 25"), the obfuscation method may not hide all original occurrence rates. However, obfuscating each single attribute also changes the original statistics of joint occurrences. Moreover, let a be the number of attributes in the data set, where each attribute a_i can have n_i different values. The number of possible joint distributions with 2 variables is calculated as:

$$|P(a_i, a_j)| = \sum_{i,j \in \mathbb{Z}_a \text{ s.t. } i \neq j} \left(n_i \times n_j\right).$$

This equation can be generalized for b variables such that each product has b terms. Note that the number of joint occurrences increases as b and n_i increase, which makes it even harder to predict the chosen b variables. In Sect. 12.2, we

provide both the single and joint probability distributions for the real data set used in our experiments and show that the proposed obfuscation method hides the joint probability distributions up to a certain level.

Theorem 4 *The privacy-preserving targeted advertising scheme provides semantic security in accordance with Definition 9.*

Proof Let the original view $v(H_n)$ and the trace $\gamma(H_n)$ be

$$v(H_n) = \{(id(1), ..., id(l)), C, \mathcal{I}, \mathcal{Q}, \texttt{sig}(\mathcal{Q})\}$$
$$\gamma(H_n) = \{(id(1), ..., id(l)), (|C_1|, ..., |C_l|), |\texttt{sig}(\mathcal{Q})|, (||Q_1||, \ldots, ||Q_n||), \kappa, ||\mathcal{I}||, S_p(H_n), A_p(H_n)\}.$$

Further let, $v^*(H_n) = \{(id^*(1), \ldots, id^*(l)), C^*, \mathcal{I}^*, \mathcal{Q}^*, \texttt{sig}(\mathcal{Q})^*\}$ be the view simulated by the simulator S.

The proposed method is semantically secure if $v(H_n)$ is indistinguishable from $v^*(H_n)$.

- The first component of the view $v(H_n)$ is the document identifiers $id(i)$, which are also available in the trace. Hence, S can trivially simulate document identifiers as $\forall i\ id^*(i) = id(i)$. Since $id^*(i) = id(i)$, they are indistinguishable.
- Each subscriber identifier is encrypted using a pseudorandomness against chosen plaintext attack (PCPA) secure encryption method (e.g., AES in CTR mode). The output of a PCPA-secure encryption method [26] is by definition indistinguishable from a random number that has the same size with the ciphertext. To simulate ciphertexts C, S assigns l random numbers to C^* such that $C^* = \{C_1^*, \ldots, C_l^*\}$ where for all i, $|C_i^*| = |C_i|$. Note that size of each ciphertext is available in the trace. Considering for all i, C_i and C_i^* are indistinguishable, C and C^* are also indistinguishable.
- The secure searchable index $\mathcal{I}$ is composed of index entries for each genuine profile and fake profiles. Note that each index entry is generated by applying "bitwise product" operation on the HMAC outputs of each attribute value in the profile. S simulates searchable index $\mathcal{I}$, by generating $||\mathcal{I}||$ index entries using the HMAC function with a random key S chooses. For each entry in simulated $\mathcal{I}^*$, S randomly selects a set of κ attributes and applies the HMAC and the reduction methods accordingly. An index entry in $\mathcal{I}^*$ is generated with exactly the same method as an entry in $\mathcal{I}$ other than the HMAC key. Since HMAC was proved to be a secure pseudorandom function (PRF) [31], an entry of $\mathcal{I}^*$ is indistinguishable from an entry in $\mathcal{I}$. Moreover, both real index and simulated index have the same number of index entries (i.e., $||\mathcal{I}|| = ||\mathcal{I}^*||$). Therefore, $\mathcal{I}$ is indistinguishable from $\mathcal{I}^*$.
- $\mathcal{Q} = \{Q_1, \ldots, Q_n\}$ is a set of n consecutive queries, where each query Q_i is a bitwise product of trapdoors for the search keywords of Q_i. $||Q_i||$ is the number of keywords in query Q_i. Note that each trapdoor is an output of the HMAC function which is a random bit string of size r, where $\frac{r}{2^d}$ bits are expected to be zero.
  Simulator S generates $\mathcal{Q}^*$ as follows. For each Q_i^*, S generates $|Q_i|$ random binary strings, where the number of zero bits is a random number with mean $\frac{r}{2^d}$ and take the bitwise product of those $||Q_i||$ strings to generate Q_i^*. The number of search terms $||Q_i||$ is available in the trace. Both Q_i and Q_i^* are bitwise products of same number of bit stings of length r with same number of expected zeros. Therefore, Q_i and Q_i^* are indistinguishable. Since for all i, Q_i and Q_i^* are indistinguishable, $\mathcal{Q}$ and $\mathcal{Q}^*$ are also indistinguishable.
- The RSA signature of a query $\texttt{sig}(Q_i)$ is a random looking number of size $|\texttt{sig}(Q_i)|$ (i.e., size of public key N) which is available in trace $\gamma(H_n)$. To simulate $\texttt{sig}(\mathcal{Q})$, S assign n random numbers of size $|\texttt{sig}(Q_i)|$ to $\texttt{sig}(\mathcal{Q})^*$. $\forall i$ $\texttt{sig}(Q_i)$ and $\texttt{sig}(Q_i)^*$ are two indistinguishable random numbers of same length. Hence, $\texttt{sig}(\mathcal{Q})$ and $\texttt{sig}(\mathcal{Q})^*$ are also indistinguishable.

The simulated view v^* is indistinguishable from genuine view v since each component of v and v^* is indistinguishable. Hence, the proposed method satisfies semantic security. □

Theorem 5 *The privacy-preserving targeted advertising scheme is privacy-preserving in accordance with Definition 10.*

Proof The proposed scheme provides semantic security, statistical privacy, trapdoor nontransferability and access control. Hence, it is a privacy-preserving targeted advertising scheme under the semantic security model. □

11 Complexity

We give the details on the complexity of the proposed technique in this section. The communication and computational costs are evaluated separately. The notation is given in Table 2. The performance evaluation presented here only serves to provide an intuition for the possible incurred cost.

11.1 Communication costs

Communication costs are given on basis of interactions between the participants in Table 3. Values inside the brackets should be taken into account when the trapdoor nontransferability protocol, given in Sect. 7.4, is used, where the RSA modulus is 1024 bits.

Table 2 Notations

N	Number of profiles
n	Number of keywords in each profile for the given rank
t	Number of obtained keywords by an advertiser
δ	Number of keywords in a (multi-keyword) query
$\mathcal{R}$	Number of profiles matched with the query
α	Number of subscribers for a specific advertisement
r	Size of an index
h	Hash digest length
η	Number of rank levels
l	Hash digest size
$E(S_j)$	Encryption of statistics for subscriber j

Table 3 Communication costs incurred by each party in the proposed system

	Communication cost (bit)
IPTV-server	$Nr\eta$ (η ranking)
	$N \cdot E(S)$
IPTV-advertiser	$tr + [1024+r]$
Advertiser-IPTV	32α (or encrypted pids) + advertisement
Advertiser-server	$r + [1024 + r]$
Server-advertiser	Response (or encrypted statistics)+[1024]

– *IPTV-Server communication.* The IPTV sends profile indexes to the server periodically (e.g., weekly), which is $Nr\eta$ bits in case η levels are used for ranking. It also sends ciphertexts in Eq. (11) (related to statistical information) for each profile to the CS.
– *IPTV-Advertiser communication.* The IPTV sends the trapdoors for the keywords to the advertiser. This communication is repeated every time the advertiser asks for new keywords or when the secret key of the IPTV is changed.
The advertiser sends pseudo-identifiers (id) of target subscribers and advertisement to the IPTV. Therefore, the advertiser sends 32α bits to the IPTV assuming that each id is a 32-bit integer, plus the content of the advertisement. In case that statistical information is used, a list of ids is encrypted by the CS and sent to the advertiser.

– **Advertiser-Server communication.** The advertiser sends an r-bit query index to the server. The server sends only an anonymized id list as a response to the advertiser. When ranking is used, η such lists are sent. In the case that statistical information is used, server sends the result of homomorphic encryption over the set of matched subscribers to the advertiser.

Note that only the actual payloads are considered in the communication costs given in Table 3, omitting the overhead data such as the message headers.

11.2 Computational costs

The computational cost for each party in the system is presented in this section.

– **IPTV,** creates profile indexes and Paillier encryption of statistics periodically; it also generates trapdoors for each advertiser.
– **Advertiser,** only prepares an index for a query which involves bitwise product of trapdoors corresponding to keywords in a conjunctive query. It also performs two Paillier decryption operations to obtain statistics of the matching profiles.
– **Server,** performs search operation, which is basically r-bit binary comparisons of query indexes with the database entries. If ranking is used, the server performs at most η-1 additional binary comparisons for each matching profile. It also performs Paillier encryption (only modular multiplications) over a set of encrypted statistics of matched subscribers.

Computational costs are summarized in Table 4. Values inside the brackets should be taken into account when the

Table 4 Computational costs incurred by each party in the proposed system

	Computational costs (bit)
IPTV	$\sum_{j=1}^{j=N} \sum_{i=1}^{i=\eta} (n \times h$-bit hash+$n$ bitwise product of r-bit string).
	[for each advertiser:
	$t \times (h$-bit hash)+t bitwise prod. of r-bit trapdoors
	1024-bit RSA signature.]
	$\sum_{j=1}^{j=N} E(S_j)$
Advertiser	r bitwise prod. of δ trapdoors (conjunctive query)
	[1024-bit RSA signature verification.
	To get response:
	RSA decryption of the symmetric key and
	symmetric decryption of the response.]
	Paillier decryption of returned results
	[1024-bit RSA signature verification.
	r binary comparisons.]
	(without ranking)
	$N \times r$ binary comparisons.
Server	(with ranking)
	$N \times r + (\eta - 1) \times \beta \times r$ binary comparisons.
	One RSA Encryption of responses.
	Paillier encryption over encrypted statistics of $\mathcal{R}$

trapdoor nontransferability protocol is used, where the RSA modulus is 1024 bits.

12 Experimental setup

We provide the results obtained from two general experimental setups. Our first experimental setup is based on the synthetic data set created by our synthetic generator, whereby we assume that the statistics of the data set is unknown to any adversary. In this setting, the query results including matching pseudo-ids are returned to the advertiser. In the second experimental setup, we apply our method on the real data of the MovieLens data set, assuming the availability of some background knowledge on the data set statistics. In the latter case, only aggregate statistical data are returned to the advertiser. All experiments are performed on a platform featuring Windows 7 machine with Intel Xeon 6 Core running at 3.2 GHz.

The indexing method that we employ covers all the information on keywords in a single r-bit index file. Independent from the hash function, after reduction and bitwise product operations, there is a possibility that index of a query may wrongly match with an irrelevant profile. Two well-known metrics for evaluating the success rate of a system are the precision and recall rates.

Precision is the ratio of the number of retrieved items that actually contains the queried terms to the number of all retrieved items. In other words, it is the ratio of correct matches to all matches.

Recall is the ratio of the number of retrieved items that actually contain the queried terms to the number of all items in the data set that contain the queried terms.

In our system, the recall rate is always one, meaning that if a profile contains all of the keywords in the query, that profile will definitely be a match to the query.

12.1 Experiments on synthetic data

The performance metrics such as efficiency and scalability are tested by generating a synthetic data set which is sufficiently large and diverse. Without loss of generality, we consider a scenario, where the profile of each subscriber consists of seven demographic features and eleven watching preferences. The demographic features are age, gender, marital status, education, occupation, city and location, whose values are the keywords in our scheme. Watching preferences enumerate the interest of an individual on a specific IPTV content, which can be categorized as follows. Morning watching preferences are marriage, news and health programs. Similarly, afternoon watching preferences are marriage, TV series and sport programs, while prime-time watching preferences are news, competition, series, talk show and sport programs.[4] Each watching preference is also taken as a keyword in the proposed solution.

Our synthetic demographic profiles are assigned randomly within the predefined categories (e.g., married, single, widow or divorced for "marital status"), and watching habits are calculated according to these demographic features. The value for a program is a real number between 0 and 1, indicating the rate of time the subscriber spends on watching the program during a week. The sum of these numbers for each week and for each subscriber must not exceed one.

We also speculate that watching habits of individuals are correlated to their demographic features. Therefore, any synthetic data should take into account such dependencies. We use our custom-made probabilistic data generator. The process conforms to some simple expectations about the types of correlations that exist between demographic features and watching preferences.

We performed experiments for the scenario where the advertiser sends single- or multi-keyword queries to find subscribers of matching profiles for an advertisement where subscribers' pseudonyms and their relevancy scores to the queries are returned to the advertiser.

In the experiments, the HMAC function produces 300 bytes output, which is generated by concatenating different length SHA2-based HMAC functions. We choose $d = 5$; therefore, after reduction, the index size (r) is 60 bytes (480 bits). The database has different number of subscribers varying from 5000 to 80,000 with each profile having at most 18 attributes (i.e., keywords).

12.1.1 Precision rate

We evaluate the success rate of the system using the precision metric. Note that the recall rate is always one in the proposed method. While this test on the synthetic data is done only to show how the number of keywords and the index entry size of the profiles affect the precision rate, the precision is also evaluated using a real data set and the actual success rate of the system is presented in Sect. 12.2.

In Fig. 4, we compare the precision values of randomly chosen queries containing one to five keywords over the synthetic data set where profiles have 18, 28, 38 and 48 attributes and index size (r) is 480 bits. The precision rate decreases rapidly after 38 attributes per profile due to the increase in the number of zeros at the index file.

In order to gain a deeper insight, we extend our experiment to test the relationship between the parameters. If more attributes are required per profile, precision can be improved by increasing the index size (i.e., choosing a longer HMAC function) and choosing a larger base d. The precision rate

[4] Afternoon sport and prime-time sport programs are independent fields.

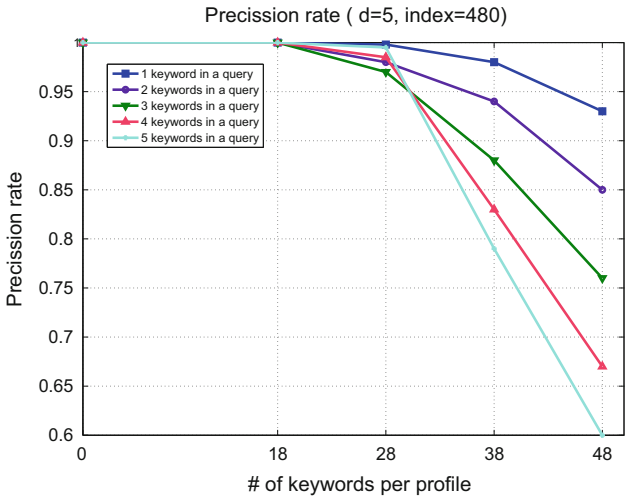

Fig. 4 Precision($d = 5$, index = 480 bits)

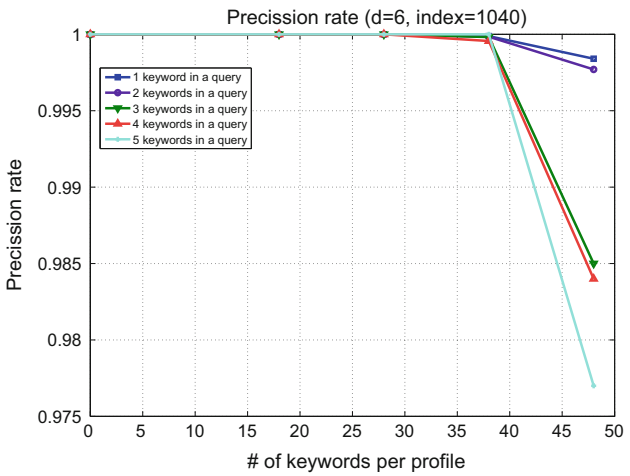

Fig. 5 Precision($d = 6$, index = 1040 bits)

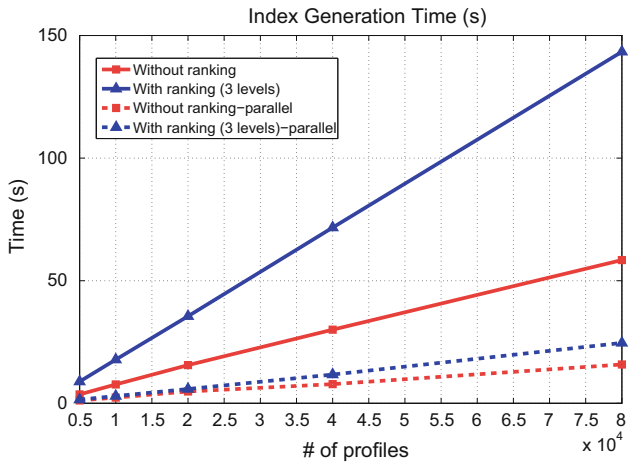

Fig. 6 Timings for index construction with 18 keywords per profile (on IPTV side)

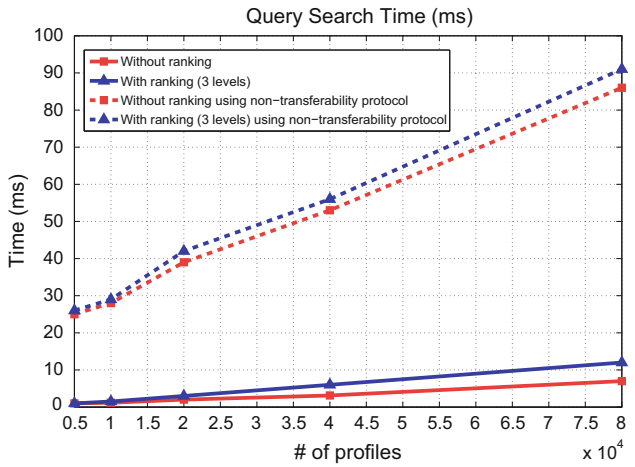

Fig. 7 Timings for query search with and without ranking (on the CS side)

given in Fig. 5 is calculated for 1040-bit index when $d = 6$. We observe a significant improvement in the precision rate compared to the rates given in Fig. 4 as the index size increases.

As the number of keywords in the profiles increases, the number of zeros in the index for profiles also increases, which increases the chance of matching these profiles incorrectly. This inevitably decreases the precision rate. However, further increase in the number of keywords in a query will result in fewer number of matches which may also have some positive effect on the precision rate.

12.1.2 Timing results for construction

Timing results for the secure index construction operations, performed on the IPTV side, are presented in Fig. 6. Note that these operations are only performed periodically (e.g., weekly) and the index construction method can be trivially parallelized. Therefore, the presented technique is practical and efficient from the perspective of the IPTV. Timing results for index generation with parallelization, utilizing all the six cores in the platform are illustrated by dashed lines in Fig. 6.

Timing results on the server side for the search operation, with and without ranking, indicate relatively low latency as shown in Fig. 7. Timings for query search when using the trapdoor nontransferability protocol are also illustrated by dashed lines in Fig. 7, which is also relatively low. Note that the signature verification operation in the protocol, which dominates the timing, does not have to be performed every time. Therefore, the scheme is also efficient from the perspective of the CS. Query generation by an advertiser takes negligible amount of time and therefore omitted here.

12.1.3 Timing results for data mining algorithms

We have performed two experiments to demonstrate the efficiency of the two most common data mining algorithms in our setting, namely clustering and content-based filtering.

Table 5 Timing results for obtaining the entire demographic data used in clustering based on demographics

Number of profiles	Advertiser total time (ms)	CS query search time (ms)	Communication time (ms)
5000	164	58	106
10,000	422	112	310
20,000	656	219	437
40,000	1345	451	894
80,000	2639	1185	1454

Table 6 Time for content-based filtering with five attributes

Number of profiles	Prediction time (ms)
5000	15
10,000	40
20,000	207
40,000	599
80,000	2249

Clustering

Advertisers can use demographic filtering [21] to categorize subscribers based on their demographics and send advertisements accordingly. Since there are seven demographic features in our data model and total number of keywords pertaining to the demographic features is 42, the advertiser sends a total of 42 different (single keyword) queries to obtain the information about subscribers' demographics. Then the advertiser can utilize the results of queries to generate a relevant segmentation for his advertising purpose. This experiment is implemented using a reliable socket communication (i.e., TCP) on the same computer.

The timing results are presented in Table 5 for different number of subscribers. While the second column lists the total elapsed time for the advertiser, the third column indicates the time the CS spends during the entire process. The last column shows the time spent on the communication. Note that the time spent for the actual clustering algorithm is not reported here.

Content-Based Filtering

The advertising agency can use content-based filtering technique [21], as described in Sect. 5.1.1, to send advertisements to related subscribers. Each advertisement can be described by attributes or characteristics of the products, which are the subject of the advertisement. The advertiser uses a similarity metric between the advertisement attributes and subscribers' watching habits or demographics to predict their potential interests in a specific advertisement.

The execution time spent by the advertiser to compute predictions of an advertisement with five attributes is illustrated in Table 6 for various data set sizes.

12.2 Experiments on real data

We also performed a number of experiments using a real data set, for the scenario where the advertiser sends queries to the CS and only statistics of results are returned to advertiser. We used the MovieLens data set [25] as the real data since there are no published IPTV data that can be used for scientific purposes. The properties of the MovieLens data set are described in Sect. 7.1.

Our experiments are conducted on movie-based and genre-based versions of the MovieLens data set separately. The genre-based data set is a relatively simple version, where subscribers are mainly profiled according to their demographics and preferences of movie genres. The movie-based implementation utilizes the full data set. The latter implementation, where there are more than 900 keywords, helps to evaluate the performance of the proposed scheme in an extreme case.

12.2.1 Genre-based implementation

In genre-based implementation, the profile of a subscriber is generated according to the genres of movies that the subscriber has rated. There are 18 genres in the MovieLens data set, and each movie may have more than one genre. To accommodate 1–5 score model of the MovieLens data set, we create five levels of index, whereby the subscribers' interests increase by each level. We employ a method to determine the genres that will be in each level of a subscriber profile as follows.

First, the frequency of genres of movies rated as i by a subscriber is calculated for each $i \in \{1, \ldots, 5\}$. Then the interest of a subscriber s for each genre is calculated as in the Eq. (19).

$$Interest_{(j,s)} = \frac{1}{\omega} \sum_{i=1}^{5} freq_{ji} \times i \tag{19}$$

Here $freq_{ji}$ is the frequency of the genre j, which is rated as i by the subscriber s. Also, j is the index of a genre, and ω is the total number of movies that is rated by the subscriber s. After calculating the interest rates of the subscriber for all genres, the interest level of genres in the subscriber's profile is set by comparing the calculated rates with the thresholds in Table 7. Note that these thresholds can be chosen in any arbitrary way.

Example 5 Let the frequencies of movie genres rated by the subscriber s_1 be as in Table 8 and total number of movies rated by this subscriber be 52. The interest rate of s_1 in $drama$ and $animation$ genres is calculated as follows:

Table 7 Thresholds for interest rate

Interest rate	Level
$1.5 \leq interest$	5
$1 \leq interest < 1.5$	4
$0.2 \leq interest < 1$	3
$0.1 \leq interest < 0.2$	2
$0 < interest < 0.1$	1

Table 8 Example of frequencies of genres given by a subscriber s_1

	Drama	Animation	Musical	War
Rate 1	0	0	0	0
Rate 2	0	0	0	0
Rate 3	0	4	2	0
Rate 4	12	8	6	0
Rate 5	9	6	6	2

$$Interest_{(drama,s_1)} = \frac{1}{52}[12 \times 4 + 9 \times 5] = 1.78$$
$$Interest_{(animation,s_1)} = \frac{1}{52}[4 \times 3 + 8 \times 4 + 6 \times 5] = 1.42 \quad (20)$$

According to the thresholds given in Table 7, $animation$ and $drama$ genres are placed in levels 4 and 5, respectively.

The proposed method also enables the advertiser to specify interest level in his queries, in addition to the keywords. For instance, the advertiser can generate a query for subscribers who are interested in $drama$ genre with degree 5.

The following figures are used in the experiments: The index size is chosen as 402 bytes, while the minimum and maximum numbers of keywords in a profile turn out to be 5 and 23, respectively. Reduction size is selected as $d = 8$. There are 6040 subscribers with 5 level scoring; thus, the number of indexes generated by the IPTV is $5 \times 6040 = 30200$. It takes 1.94 m to generate all the indexes and 30 s to generate encrypted statistics in Eqs. (11) and (12). Using parallelization, the index generation time is reduced to 38 s using six cores. Time for generating a query is negligible and both precision and recall rates for the queries are found to be 1.

Recall that the IPTV obfuscates the data by adding some fake profiles as explained in Sect. 7.5. Whenever the CS searches for a query over encrypted data, the fake profiles are also included in the query results, but only genuine profiles are revealed to the advertiser. We select genre-based data set to insert fake profiles since it contains fewer number of keywords in each profile.

When generating fake profiles, the aim is to eliminate statistical bias for the keywords, where their occurrence frequencies in the data set are known as background knowledge. For this, we select a target interval for the frequency of a keyword and add fake profiles until it reaches in this interval. In our experiments, after inserting 12000 fake profiles, frequencies of all keywords for each attribute (Age, $Gender$, $Occupation$ and $Genre$) are brought into the targeted interval. The probability distribution of keywords before and after inserting fake profiles is demonstrated in Figs. 8, 9 and 10. In our setting, due to efficiency reasons, we obfuscate age groups and gender in one group, occupations and genres in other groups. Hence, analyzing the search results, an adversary may learn whether the query is for an occupation, a genre or something else but cannot learn anything other than this.

For example, Fig. 8 illustrates that all keywords related to gender and age have almost the same frequency after fake profile insertion. It is difficult to distinguish a male from female using the statistics. Similarly, probability distributions of all keywords pertaining to occupation fall in the interval between 0 and 0.1 as can be observed in Fig. 9. A similar case also exists in the keywords for the movie genres as shown in Fig. 10.

The provided obfuscation method ensures that the frequency of each single attribute is within a security range. The adversary model that has the information of all joint probability distributions is beyond the scope of this work. Nevertheless, the insertion of fake profiles also perturbs the joint distributions and those statistics can still be hidden with this method up to a certain level. Note that the adversary cannot trivially know the attributes used in the query. Let there be a attributes in the data set, where each attribute a_i can have n_i different values. The number of joint probability distributions with 2 variables is calculated using the function given in Sect. 10 as:

$$|P(a_i, a_j)| = \forall i, j \in Z_a \text{ s.t. } i \neq j \sum (n_i \times n_j).$$

We analyze the effect of the proposed obfuscation method on the joint probability distribution, using the MovieLens data set. In the data set, there are four attributes (Age, $Gender$, $Occupation$ and $Genre$) that have 2, 4, 21 and 18 values, respectively. Hence, the number of joint probability distributions with 2 variables is 620. Among those 620 joint probabilities with 2 variables, after applying obfuscation, the change in 486 of the probabilities is greater than 1 %. The average change is calculated as 2.2 % where the maximum change is 26 %. These experimental results show that it is not trivial to learn the keywords (i.e., attribute values) in queries by correlating the statistics of the original data set with the query results of the obfuscated data set.

The analysis do not consider the case of joint distributions for three or more attributes, but it is important to note that the number of possible joint distributions further increases as the number of variables used gets larger. Moreover, any joint probability distribution can further be obfuscated by increas-

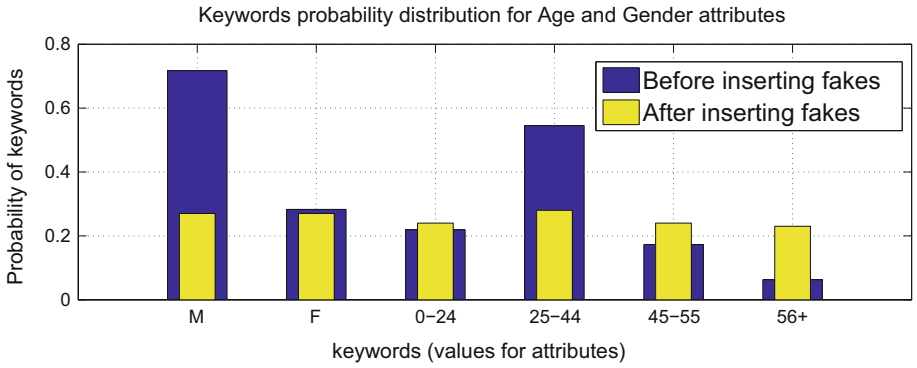

Fig. 8 Probability distribution of values for age and gender attributes

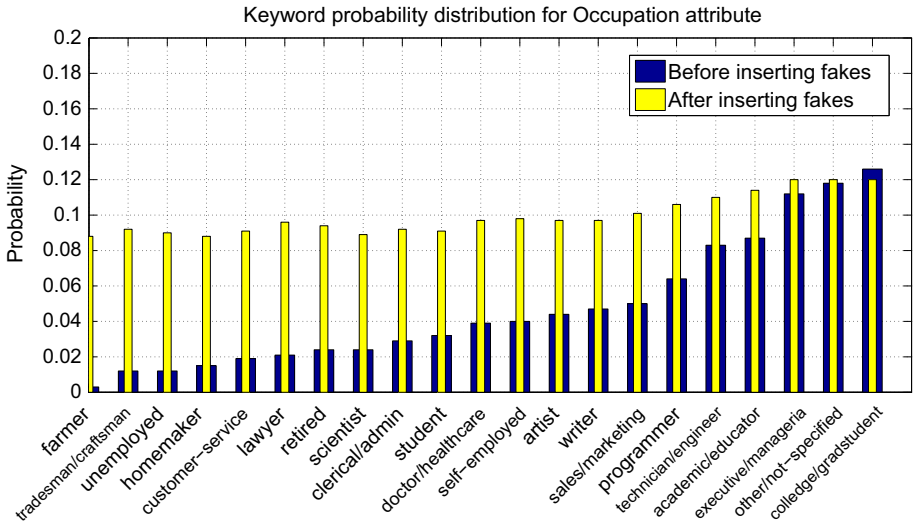

Fig. 9 Probability distribution of values for occupation attribute

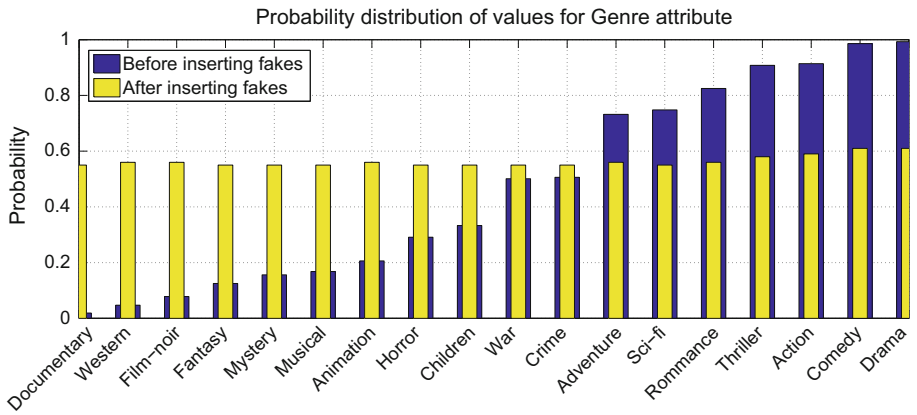

Fig. 10 Probability distribution of values for genres attribute

ing the number of fake profiles. However, there is a trade-off between privacy and efficiency. The index generation time and index size will both increase as number of fake profiles increases.

In addition, the number of zeros in an index is a direct function of the number of keywords in the corresponding profile. Therefore, the CS may identify fake profiles in the data set by inspecting the number of zeros in their indexes. In order to prevent such an attack, keyword distribution (i.e., number of keywords in each profile) in fake profiles is set statistically indistinguishable from those in genuine ones. Figure 11 demonstrates keyword distributions in indexes of fake and genuine profiles. The Y axis represents the percentage of genuine and fake subscribers whose profiles contain certain number of keywords as indicated in the X axis. The figure shows that there is not a statistically significant difference between the number of keywords in genuine and fake profiles; thus, it is hard to determine whether a given profile is fake or genuine.

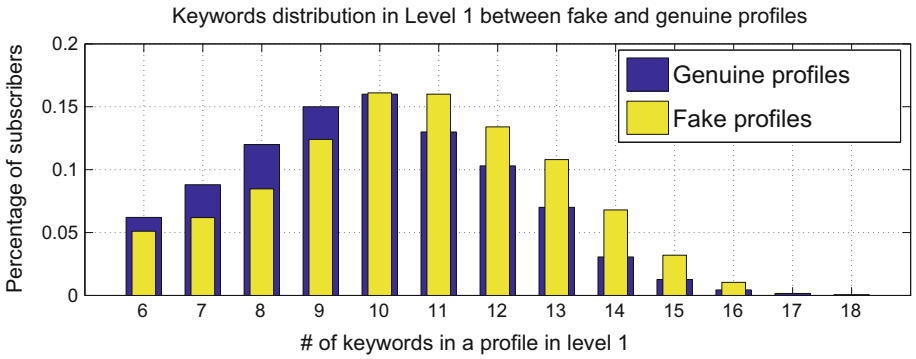

Fig. 11 Keywords distribution in level 1 between fake and genuine profiles

The timing results for fake profile generation show that it is acceptable for practical usage. Using parallelization, it takes 1.93 m to generate all indexes (i.e., genuine+fake) and 71 s to generate all statistics. Query generation takes negligible amount of time.

12.2.2 Movie-based implementation

In the movie-based implementation, a profile of a subscriber is generated using the movies rated by him and his demographics. We use a five-level index to accommodate the 1–5 rating scores used in the MovieLens data set. Whenever a query contains multiple movies in a single query, only subscribers who give the same rate to those movies will be returned as a result. Therefore, sending one-keyword queries is better for the accuracy. This way it can easily categorize subscribers by their ratings to a movie. Index size is chosen as 1202 bytes, which is longer than the previous case, to provide accuracy for profiles that contain as many as 950 keywords. The minimum and maximum numbers of keywords are 5 and 950, respectively. Reduction size is set as $d = 8$.

The number of indexes that are generated by the IPTV is $5 \times 6040 = 30200$ (5 level rank with 6040 subscribers). It takes about 30 m to generate all indexes and 40 s to generate all statistics. As in all the previous cases, the time it takes to generate a query is negligible. Using parallelization on a six-core computer, we can reduce the index generation time approximately six times (reduced to 5 m). Note that the index generation shows strong scalability, provided a sufficient number of cores.

In Table 9, we present the precision rates which are calculated by taking average of 250 random queries when the number of keywords in a query varies from 1 to 7. Note that precision for one-keyword queries is quite high (about 0.922). The precision goes up to 1 with the increase in the number of keywords in a query. The main reason is that when the number of queried keywords is high, it is hard to find any real match in the data set whose profile contains all the keywords in the query. We also enumerate the average search times spent on queries with different number of keywords in Table 9.

Table 9 Precision rate and search time for a query

# of keywords in a query	Precision rate	Search time (ms)
1	0.922	71
2	0.77	58
3	0.80	55
4	0.90	54
5	0.96	54
6	0.98	53
7	1	53

Collaborative Filtering

Consider a scenario where the advertiser wants to send an advertisement for the movie m_4 to subscribers $\mathcal{M}_1$ who have watched the movies $\{m_1, m_2, m_3\}$, but not m_4. For finding appropriate subscribers, he first finds the subscribers $\mathcal{M}_2$ who have watched all four movies $\{m_1, m_2, m_3, m_4\}$. Then, computes a prediction for each subscriber $c \in \mathcal{M}_1$, by calculating the correlation between c and each subscriber in $\mathcal{M}_2$.

Example 6 Suppose that $\{m_1, \ldots, m_4\}$ are movies "Liar liar," "Brazil," "Barcelona" and "Smoke" correspondingly, which are chosen from the *MovieLens* data set. The advertiser wants to recommend the movie "Smoke" to subscribers who watched the other three movies. It turns out that there are 29 subscribers in the *MovieLens* data set who have watched the three movies and 16 of them have also watched the fourth movie. Thus, the advertiser computes predictions for the other 13 subscribers who have not watched the fourth movie yet. Table 10 shows predictions which are calculated using the method described in Sect. 9. It takes the CS only 303 ms to compute predictions for the 13 subscribers for the content m_4. The advertiser can also select a threshold value to send the advertisement to subscribers whose predictions are greater than a given threshold.

The experiments in this section confirm that the proposed method can be efficiently used even for extreme cases, where the number of keywords in each profile is high.

Table 10 Prediction for movie m_4

SubscriberID	m_1	m_2	m_3	m_4
151	5	4	4	3.40
624	2	5	5	3.20
1371	4	3	4	2.71
1983	5	4	5	3.70
2092	5	2	4	2.88
3054	5	2	3	2.58
3067	5	3	3	2.81
3128	5	4	5	2.49
3292	5	4	5	3.70
3371	5	2	3	3.11
4478	5	4	5	2.58
4728	4	4	3	2.750
4933	5	5	4	3.74

13 Conclusion and future work

In this paper, we propose a practical, secure and privacy-preserving targeted advertising service for the IPTV subscribers. The proposed method can be implemented over a cloud service due to the relaxed, semi-trusted assumption on the server.

While the privacy of subscribers is protected, the advertisements can be targeted to prospective customers with accuracy since precision of the method is quite high and recall is 1 (i.e., perfect recall). The rating mechanism adopted in our solution targets only subscribers who demonstrate desired relevance to queries. Furthermore, advertising agencies can keep their strategies for reaching potential customers unexposed to the cloud server or other advertisers.

From the IPTV's perspective, the proposed system has many advantages. It allows outsourcing all the services pertaining to advertisement without any adverse effect on the security and privacy issues concerning its subscribers. Furthermore, advertisers (or any other party who has access to the database on the cloud) can utilize the data only if they are authorized to do so. The IPTV can control the access to the database in a fine-grained manner in the sense that the advertiser can only access to the part of the database related to keywords, for which he holds trapdoors.

The idea of returning statistics on aggregate data, instead of fixed pseudonyms as query response, is useful when some background information about the subscribers is available. Without the proposed obfuscation method, analyzing the background statistics may reveal the identity of subscriber from the returned pseudonyms. Whenever the statistics might disclose sensitive information, the flexibility in our system allows the cloud server to return only partial information. The fake subscribers that are inserted for obfuscating the data set prevent the cloud server from identifying a specific subscriber and/or learning any extra sensitive information about him. The cloud server cannot distinguish genuine subscribers from the fake ones which can match with queries and be only identified by either the advertiser or the IPTV.

The algorithms used in index generation and query execution phases demonstrate strong scalability in terms of parallelization. The experimental results show that speedups proportional to the number of cores employed can be achieved for both index generation and query execution operations. This suggests that the proposed algorithms are scalable and can efficiently be used even with large data sets.

Lastly, the proposed system can be utilized in a recommendation system as well. From a general perspective, in our solution, the advertisement can be any product, service or IPTV content (e.g., a movie, sitcom).

As a future work, we intend to adapt the proposed scheme for Big Data applications, in which the size of the data set and the data velocity are both very high. A possible solution method can be utilizing extreme levels of parallelism on a cluster of computers.

Acknowledgments This work was supported by Turk Telekom under Grant Number 3014-02.

References

1. Khayati, L.J., Savas, E., Ustaoglu, B., Orencik, C.: Privacy-preserving targeted advertising scheme for iptv using the cloud. In: SECRYPT, pp. 74–83 (2012)
2. Kodialam, M., Lakshman, T., Mukherjee, S., Wang, L.: Online scheduling of targeted advertisements for iptv. In: 2010 Proceedings IEE, INFOCOM, pp. 1–9 (2010)
3. Min, W.H., Cheong, Y.G.: An interactive-content technique based approach to generating personalized advertisement for privacy protection. HCI **9**, 185–191 (2009)
4. Kraus, J.: "Iptv privacy risks." http://www.cedmagazine.com/articles/2006/02/iptv-privacy-risks, February 2006. Retrieved 02 April (2013)
5. Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R.H., Konwinski, A., Lee, G., Patterson, D.A., Rabkin, A., Stoica, I., Zaharia, M.: Above the clouds: a berkeley view of cloud computing. In: Tech. Rep. UCB/EECS-2009-28, EECS Department, University of California, Berkeley (2009)
6. "Opinion 05/2012 on cloud computing." http://ec.europa.eu/justice/data-protection/article-29/documentation/opinion-recommendation/files/2012/wp196_en.pdf, July 2012. Retrieved 02 April (2013)
7. Kamara, S., Lauter, K.: Cryptographic cloud storage. In: Proceedings of the 14th International Conference on Financial Cryptography and Data Security, FC'10, pp. 136–149. Springer, Berlin (2010)
8. Sweeney, L.: K-anonymity: a model for protecting privacy. Int. J. Uncertain. Fuzziness Knowl. Based Syst. **10**, 557–570 (2002)
9. Boneh, D., Crescenzo, G.D., Ostrovsky, R., Persiano, G.: Public key encryption with keyword searches. In: Proceedings of Eurocrypt 2004. LNCS, vol. 3027, pp. 506–522 (2004)

10. Wang, C., Cao, N., Li, J., Ren, K., Lou, W.: Secure ranked keyword search over encrypted cloud data. In: ICDCS'10, pp. 253–262 (2010)
11. Cao, N., Wang, C., Li, M., Ren, K., Lou, W.: Privacy-preserving multi-keyword ranked search over encrypted cloud data. In: IEEE INFOCOM (2011)
12. Boneh, D., Waters, B.: Conjunctive, subset, and range queries on encrypted data. In: Theory of Cryptography, vol. 4392 of Lecture Notes in Computer Science, pp. 535–554, Springer, Berlin (2007)
13. Ballard, L., Kamara, S., Monrose, F.: Achieving efficient conjunctive keyword searches over encrypted data. In: vol. 3873, pp. 414–426. Springer, Berlin (2005)
14. Shen, E., Shi, E., Waters, B.: Predicate privacy in encryption systems. In: TCC '09: Proceedings of the 6th Theory of Cryptography Conference on Theory of Cryptography, pp. 457–473. Springer, Berlin (2009)
15. Cash, D., Jarecki, S., Jutla, C., Krawczyk, H., Rou, M.-C., Steiner, M.: Highly-scalable searchable symmetric encryption with support for boolean queries. In: Canetti, R., Garay, J. (eds.) Advances in Cryptology CRYPTO 2013, vol. 8042 of Lecture Notes in Computer Science, pp. 353–373. Springer Berlin (2013)
16. Wang, P., Wang, H., Pieprzyk, J.: An efficient scheme of common secure indices for conjunctive keyword-based retrieval on encrypted data. In: Chung, K.-I., Sohn, K., Yung, M. (eds.) Information Security Applications, vol. 5379 of LNCS, pp. 145–159. Springer, Berlin (2009)
17. Orencik, C., Savas, E.: An efficient privacy-preserving multi-keyword search over encrypted cloud data with ranking. Distrib. Parallel Databases **32**(1), 119–160 (2014)
18. el Diehn, D., Abou-Tair, I., Köster, I., Höfke, K.: Security and privacy requirements in interactive tv. Multimed. Syst. **17**, 393–408 (2011)
19. Fontaine, C., Galand, F.: A survey of homomorphic encryption for nonspecialists. EURASIP J. Inf. Secur. **2007**, 15:1–15:15 (2007)
20. "Eu directive 95/46/ec - the data protection directive." http://www.dataprotection.ie/docs/EU-Directive-95-46-EC--Chapter-1/92.htm, October 1995. Retrieved January 02, (2014)
21. Aïmeur, E., Brassard, G., Fernandez, J., Mani, : Alambic : a privacy-preserving recommender system for electronic commerce. Int. J. Inf. Secur. **7**(5), 307–334 (2008)
22. Su, X., Khoshgoftaar, T.M.: A survey of collaborative filtering techniques. Adv. Artif. Intell. **2009**, 4:2–4:2 (2009)
23. Herlocker, J., Konstan, J.A., Riedl, J.: An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. Inf. Retr. **5**, 287–310 (2002)
24. CAT, "http://anony-toolkit.sourceforge.net/" (2009)
25. MovieLens, "http://www.grouplens.org/node/12."
26. Curtmola, R., Garay, J., Kamara, S., Ostrovsky, R.: Searchable symmetric encryption: improved definitions and efficient constructions. In: Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS '06, pp. 79–88, ACM (2006)
27. Bellare, M., Canetti, R., Krawczyk, H.: Keying hash functions for message authentication. In: Advances in Cryptology CRYPTO 96, vol. 1109 of Lecture Notes in Computer Science, pp. 1–15, Springer, Berlin (1996)
28. Narayanan, A., Shmatikov, V.: Robust de-anonymization of large sparse datasets. In: Proceedings of the IEEE Symposium on Security and Privacy, pp. 111–125. IEEE, Los Alamitos, CA (2008)
29. Rappe, D.: Homomorphic cryptosystems and their applications. PhD thesis, University of Dortmund, Dortmund (2004)
30. Paillier, P., Pointcheval, D.: Efficient public-key cryptosystems provably secure against active adversaries. In: Lam, K.-Y., Okamoto, E., Xing, C. (eds.) Advances in Cryptology—ASIACRYPT'99, vol. 1716 of Lecture Notes in Computer Science, pp. 165–179, Springer, Berlin (1999)
31. Bellare, M.: New proofs for NMAC and HMAC: Security without collision-resistance. In: Proceedings of the 26th Annual International Conference on Advances in Cryptology, CRYPTO'06, pp. 602–619. Springer, Berlin (2006)