IJCAT - International Journal of Computing and Technology, Volume 2, Issue 8, August 2015
ISSN : 2348 - 6090
www.IJCAT.org

Implementation of Privacy-Preserved Personalized Web Search based on Fully Homomorphic Encryption over Integers

[1] Akhila G S, [2] Prasanth R S

[1] Department of Computer Science and Engineering, Mohandas College of Engineering and Technology, Anad Thiruvananthapuaram, Kerala 695544, India

[2] Assistant Professor, Department of Computer Science and Engineering, Mohandas College of Engineering and Technology, Anad Thiruvananthapuaram, Kerala 695544, India

Abstract - Using Personalized Web Search (PWS) we can improve the quality of search results in the Internet. The existing UPS based Personalized Web Searching has many drawbacks. First, there may be a chance of eavesdropping when generalized profile forwarded to the server. Second, web server is vulnerable to web attacks like URL manipulation attacks. The impact of these attacks will affect user's personal information. So we introduce a new framework called UPES. Here, the data stored in the server-side and request from user will be in encrypted form. Fully Homomorphic Encryption over Integers (FHEI) is used for encrypting data. The experimental results show that this framework functioned in the best possible manner with the least waste of time and effort.

***Keywords* – Personalized Web Search, UPS, User Profile, Generalized User Profile, FHEI.**

1. Introduction

Personalized Web Search (PWS) is related to Web mining. Web mining is mining of data related to World Wide Web. It is divided into different categories like Web content mining, Web structure mining and Web usage mining. This PWS comes under the Web content mining. Web content mining can be thought of as extending the work performed by basic search engines. When same query submitted by different users, typical search engines return the same results regardless of who submitted the query. Here, there is no role for the user. Typically, each user has different information needed for his/her query. Therefore, the search results should be adapted to user with different information needs. Hence, introduces a new concept known as Personalized Web Search. PWS is a general category of search technique to improve the search quality based on individual user needs. Now we have different types of search engines like Google, Yahoo!, Bing etc. But, the best search engine which supports PWS is Google. If a user creates a Google account, then a user profile is automatically created at the server side. When user search through his/her account, the search engine returns the personalized search results after analysing the user profile of this particular user.

A user profile contains the personal information or interests of a particular person. Different profiling techniques are available to construct the user profile [1]. Before the user profile construction a system needs to identify the interests of users. The sources we have used in constructing a user's profile are: bookmarks from a social bookmarking site, web communities, blogs of interests etc. The first step in the construction of user profile is pre-processing. The pre-processing step involves stop word removal and stemming. These are then converted to feature vectors where the features are the terms in the documents after the pre-processing step. After performing any clustering algorithm, we get several clusters and clusters would represent interests. So if we assign weightages to interest vectors on the basis of documents downloaded and browsed we get a fairer representation of a user's current interest. The weightages are calculated based on the number of documents assigned to each cluster. So user profile has very important role in effectiveness of search quality.

The rest of this paper is organized as follows: Section 2 reviews the related works. Section 3 is about a broad introduction to Homomorphic encryption. Section 4 is about the proposed approach to solve the problem by PWS using Homomorphic encryption. Section 5 further discusses the implementation of UPES. The experimental results and findings are reported in Section 6. Finally, Section 7 concludes the paper.

2. Related Works

A. Pretschner and S. Gauch [2] proposed personalized web search based on ontology. In this technique the authors created user profile by analyzing surfed pages to identify their content and the time that was spent on it. When pages about certain subjects are visited again and again, this is taken as an indication of the user's interest in that subject. The main advantage of this approach was that except for the act of surfing, no user interaction with this system is necessary. But disclose the user private information was the main problem in this approach. Because here the user profile created based on user's surfed pages and created user profile could viewed by publically accessible search engine. L. Fitzpatrick and M. Dent [3] developed personalized web search, in this users have to register personal information such as their interests, age, and so on during the training period. Another one method was users have to provide feedback on relevant or irrelevant judgements, by rating on a scale from 1 to 5. Here 1 indicates very bad and 5 indicates very good. This approach had a lot of limitations. Explicit construction of user profiles has several drawbacks. Sometimes user provides inconsistent or incorrect information, it affect the construction of user profile. It was a time consuming process.

K. Sugiyama, K. Hatano, and M. Yoshikawa [4] suggested an adaptive web search based on user profile. The main advantage of this approach was user profile constructed without any effort or feedback from user. The main problem in previous approach was need continuous user interaction. This technique solved that problem. In this system, when a user submits a query to a search engine via web browser, the search engine returns search results corresponding to the query. Based on the search results, the user may select a web page in an attempt to satisfy his/her information need. In addition, the user may access more web pages by following the hyperlinks on his/her selected web pages and continue to browse. The proposed system monitors the user's browsing history and updates his/her profile whenever his/her browsing page changes. When the user submits a query the next time, the search results adapt based on his/her user profile. Here also the main problem was discloses of user privacy. It gave better performance than previous techniques, but the created user profile completely accessible by search engine.

M. Spertta and S. Gach [5] recommended personalized web search technique based on user's search history. In this approach, the authors constructed user profile by analysing user's search history. Search engines index millions of documents on the Internet and allow users to enter keywords to retrieve documents that contain these keywords. Browsing is usually done by clicking through a hierarchy of subjects until the area of interest has been reached. In this approach is based on building user profiles based on the user's interactions with a particular search engine. For this purpose, the developers implemented GoogleWrapper. A wrapper around the Google search engine, that logs the queries, search results, and clicks on a per user basis. This information was then used to create user profiles. A wrapper for Google that implicitly collects information's from users. Users register with their email addresses in order to create a cookie storing their user id on their local machines. When user submits a query, GoogleWrapper logs the query and the userID and then forwards the query to the Google search engine. The search engine returns result back to the user based on the created user profile of particular userID. To keep users secrets was the major problem in this approach.

Y. Xu, K. Wang, B. Zhang, and Z. Chen [6] proposed a privacy-enhancing personalized web search. Users are uncomfortable with exposing private preference information to search engines. In this approach, authors introduce an algorithm which automatically builds a hierarchical user profile that represents the user's implicit personal interests. Only portions of the user profile will be exposed to the search engine in accordance with a user's own privacy settings. A search engine wrapper is developed on the server side to incorporate a partial user profile with the results returned from a search engine. Rankings from both partial user profiles and search engine results are combined. The customized results are delivered to the user by the wrapper. In this approach, user profile kept as exposed plus private. That is why user's privacy could be preserves somewhat. This profile-based PWS does not support runtime profiling. A user profile is typically generalized for only once offline, and used to personalize all queries from a same user without making distinctions. Here ranking is take place based on only exposed user profile. L.Shou et.al[7] introduced a framework called UPS (User customizable Privacy-preserving Search). This consists of a non-trusty search

engine server and a number of clients. Here the users can customize their privacy requirements. The main component of this framework is an online profiler implemented as a search proxy running on the client machine itself. This proxy maintains both the complete user profile and the user specified privacy requirements represented as a set of sensitive nodes. For each user, the framework works in two phases, namely the offline and online. During the offline phase, a hierarchical user profile is constructed and customized with the user specified privacy requirements. Here the user profile is created based on the user's browsing history and a data set, called WordNet. The WordNet is a huge topic hierarchy covering entire topic domain of human knowledge. It is a public accessible data set.

By using this data set, the UPS could solve the problem "one profile fits all strategy". During the online phase, when user submits a query, the profiler generates a user profile in runtime in the light of submitted query. The output of this step is a generalized profile which satisfies all the privacy requirements of user. Then, the query and the generalized profile are sent together to the server for personalized search. The search results are personalized with the user profile and forwarded to the query proxy. Finally, the proxy presents the raw results to the user.

3. Homomorphic Encryption

Homomorphic encryption is a form of encryption which allows specific types of computations to be carried out on cipher text and generate an encrypted result which, when decrypted, matches the results of operations performed on the plaintext. It permits computing on encrypted data. The client can encrypt his data x and send the encryption Enc(x) to the server. The server can then take the cipher text Enc(x) and evaluate a function f on the underlying x obtaining the encrypted result Enc(f(x)).

The client can decrypt this result achieving the wanted functionality, but the server learns nothing about the data that he computed on. The Fully Homomorphic Encryption over the Integers (FHEI) scheme [8] is Homomorphic for addition and multiplication. On the basis of homomorphism property, the encryption scheme can be described as four stages: KeyGen, Encrypt, Evaluate, and Decrypt.

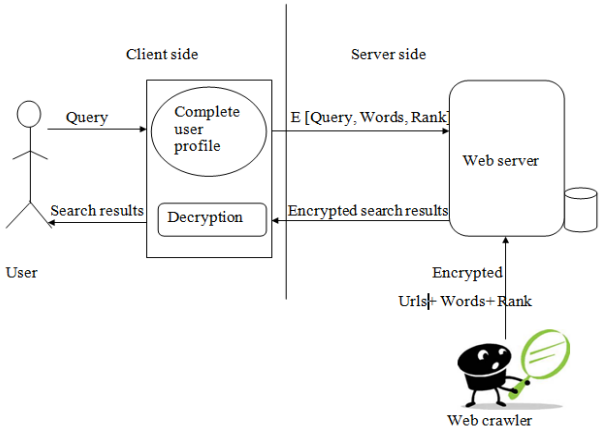

Fig. 1 Architecture of UPES.

- KeyGen: The secret key is an odd n-bit number randomly selected from the interval $[2^{n-1},2^n)$. The public key is pq+x, where q denotes the multiple parameter, and x denotes the noise to achieve proximity against brute-force attacks. Where $x=2^{2*n}$. The secret key is used for encryption and the public key is used for decryption.
- Encrypt: Cipher text, c=pq+2x+m, where m denotes the plaintext of the integer.
- Evaluate: Apply all binary addition and multiplication gates to the cipher text, perform all necessary operations, and then return the resulting integer X.
- Decrypt: Output $m' = (X \bmod p) \bmod x$.

4. Proposed System

From the literature survey it is concluded that most of the PWS system primarily focused on to improve search quality. There is no security for user's personal information. The last paper mentioned in the literature survey, tried to solve almost all problems in the PWS. But there may be a chance of eavesdropping when generalized profile forwarded to the server. Based on generalized profile, the attacker will attempt to touch the sensitive nodes of the user by recovering the segments hidden from the original user profile, and computing a confidence for each recovered topic, relying on the background knowledge in the publicly available taxonomy repository. Besides these problems, currently the web servers face the major problem is different types of web attacks like, URL manipulation attacks, trial and error attacks, directory traversal attacks etc.,.

A successful attack can have consequences like website defacement, stolen information, modification of data, and particularly modification of users' personal data, and web

server intrusion. The above problems are addressed in our UPES (literally for User Privacy-preserving Encrypted Search) framework. As illustrated in Fig.1, we have a web crawler for crawling the web pages. In normal case, the crawled details like urls, words in urls, and corresponding rank are stored on server in plain form. But here, we store these details as encrypted form. So we can protect the web server from all types of web attacks. When the user submit a query, the server will get, the encrypted combination of submitted query, related words of this term from the user profile, and also the corresponding ranks of each term in the user profile. In Section 5, we are describing the creation of user profile. Request from user to the server is in the encrypted form, so we can protect the user's personal information and avoid eavesdropping problem. For encryption, here we use Homomorphic encryption, described in Section 3. The user will get the results after decrypting the personalized search results from server. Thus we can protect user's privacy and web server from all types of attacks.

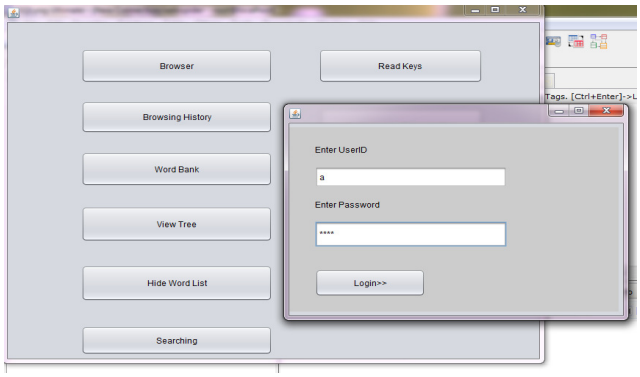

Fig. 2 User login form for accessing search interface.

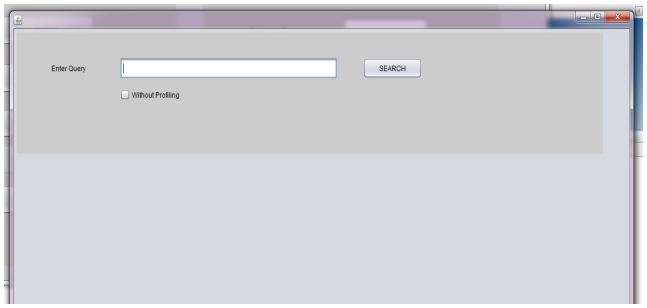

Fig. 3 Search interface.

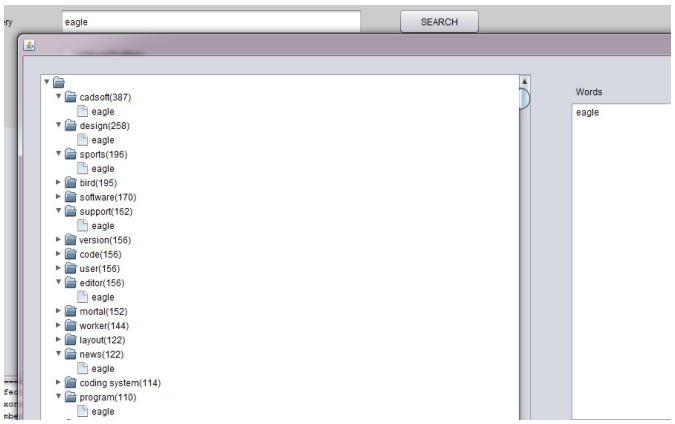

Fig. 4 Generalized user profile.

4.1 User Profile

In UPES, each user profile adopts a hierarchical structure. Moreover, our profile is constructed based on the availability of a public accessible taxonomy that is *WordNet*. For constructing the user profile we need to track the user's browsing history.

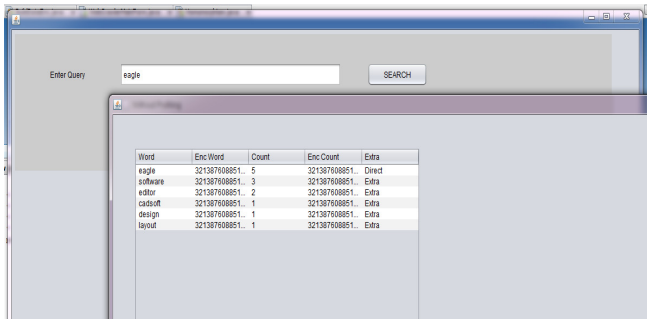

Fig. 5 User request format.

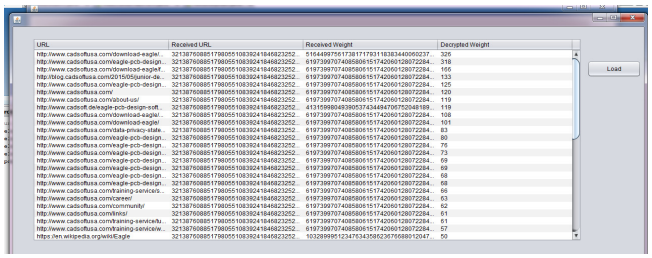

Fig. 6 Personalized search results.

Each click on the links the corresponding URL information's are stored in the database. Using these information and *WordNet* information, we can construct user profile. In section 5, we are presented the details about this step.

4.2 Online Profiler

This is an online process. In this module, user wants to enter his/her user id and password. That password is already given by web crawler while creating the user profile. Now we can enter the search query in the search interface. The submitted query, related words from user profile, and corresponding ranks of each topic from the user profile are encrypted and transferred to the web server. The operation of the web server is explained in the section 5. After performing these operations, server returns the encrypted form of personalized search results to the online profiler. Finally the online profiler presents the decrypted results to the user.

4.3 Web Crawler

A web crawler starts with a list of URLs to visit, called "seeds". As the crawler visits these URLs, it identifies all the hyperlinks in the page and adds them to the list of URLs to visit, called crawl frontier. URLs from the frontier are recursively visited according to a set of policies. Then these details are encrypted and send to the server. Here we provide an authentication for the server. Each user must register to the web crawler before accessing to the server.

4.4 Web Server

The web server has URLs, words and corresponding ranks in encrypted form. So, there is no chance for web attacks like URL manipulation, trial and error attacks etc. When the user submit a query, the server will get, the encrypted combination of submitted query, related words of this term from the user profile, and also the corresponding ranks of each term in the user profile. Since, this is in encrypted form; there is no chance for eavesdropping. The server identifies URLs for submitted query and re ranks them based on related words and ranks in user profile. Therefore, the user will get all URLs related to the submitted query and top all links from the user's interested area. Using this framework, user will get all links related to different area for the submitted query.

Table 1: Sample server database

URLs	**word**	**rank**
E(http://www.cadsoftusa.com/)	E(cadsoft)	E(7)
E(http://www.cadsoftusa.com/)	E(privacy)	E(4)
E(http://www.cadsoftusa.com/eagle-pcb-design-software/eagle-pcb-design-software/)	E(cadsoft)	E(4)
E(http://www.cadsoftusa.com/eagle-pcb-design-software/eagle-pcb-design-software/)	E(design)	E(8)

Table 2: Sample user request

word	**rank**
E(Eagle)	E(5)
E(design)	E(7)
E(cadsoft)	E(4)
E(support)	E(2)
E(privacy)	E(2)
E(user)	E(1)

5. Implementation

The UPES framework is implemented on a PC with an Intel Core i5 2.67-GHz CPU and 4-GB main memory, running Microsoft Windows 7. All the algorithms are implemented in Java.

The topic repository uses the WordNet. First step in our thesis work was download dataset from web. Here we provide an authentication for server. For each user, they need to authenticate their identity to the server, before accessing the server.

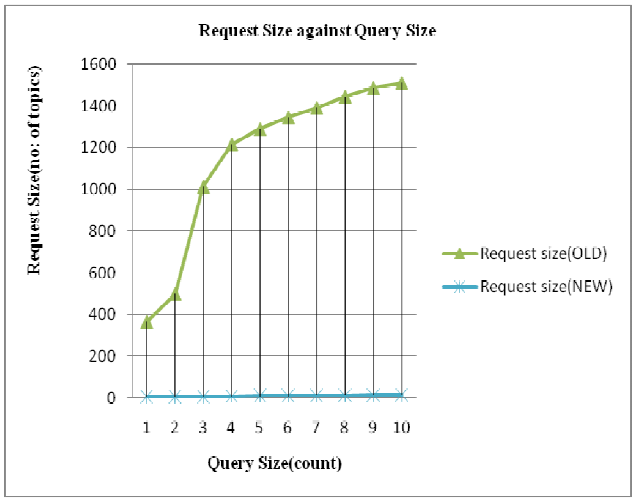

Fig. 7 Comparison of Request Size against Query Size.

We can divide the overall procedure of the thesis into three modules.

Profile construction: This is an offline process for identifying user's interests, for constructing user profile that we need to track the user's browsing history. For each click on the links, the corresponding URL information stored into the table named "*urlinfo*". Internally, we calculate the most frequent words in each URL. Means, after performs stemming process, count each word, if the

count is more than predefined value it will be stored into the corresponding entry in the table. These words are also stored into the table "*allwords*". Now we have frequent words, which might be the interested topics of the user, also we need to find corresponding related words in the WordNet. For that we use "*allwords*" table's information and dataset. Trace all related words and store those words into "*related_words*" table. Based on all these tables we constructed the user profile in a hierarchical structure. We used simple tree construction java code for the hierarchical structure.

Web crawling: A web crawler starts with a list of URLs to visit, called seeds. As the crawler visits these URLs, it identifies all the hyperlinks in the page and adds them to the list of URLs to visit, called crawl frontier. URLs from the frontier are recursively visited according to a set of policies. After getting a set of URLs, crawler sends these URLs and words in those URLs to the server after encryption.

Online profiler: Fig.2 shows the login form for user where user need to enter their user id and password, which is already given by web crawler while creating user profile. Now, user can enter the search query in the search interface. Fig.3 shows the search interface. The submitted query, related words from user profile, and corresponding ranks in the search interface is encrypted and transferred to web server.

Table 3: Comparison of Request Size against Query Size

Query size (count)	**Request size(OLD) (no: of topics)**	**Request size(NEW) (no: of topics)**
1	366	6
2	500	7
3	1013	8
4	1216	9
5	1289	10
6	1345	11
7	1391	12
8	1445	13
9	1486	14
10	1510	15

Fig.4 shows the generalized profile for the query "eagle". In the previous method, query and this generalized profile are sent together to the server for personalized search results. Here, it passes only top five terms in the generalized user profile. So, the query, five topics and their corresponding ranks in user profile are encrypted and sent together to the web server. Fig.5 shows the user request format, where the topics and ranks are in encrypted form. The profiler gets personalized search results in encrypted format and presents the results to the user after performing decryption. Fig.6 shows the personalized search results after decryption.

Web server: The operation of web server is described with an example. Suppose Table.1 shows the sample server database which is in encrypted form. Similarly, Table.2 shown sample user request which is also in encrypted form. For example, if the user submitted the query "eagle", then the corresponding request contains six terms and their respective ranks in the user profile. Here, the six terms are, the first one is submitted query and the rest are related terms of submitted query from user profile. Now, server finds the personalized search results by calculating the preference of each relevant links of submitted query from server database. For example, the word "E(design)" is present in the server database corresponding to the link "E(http://www.cadsoftusa.com/eagle-pcb-design-software/eagle-pcb-design-software/).
This link also contains the word "E(cadsoft)". So, the preference of this particular link for this submitted query can be calculated by,

Pref(E(http://www.cadsoftusa.com/eagle-pcb-design-software/eagle-pcb-design-software/)) = E(7)*E(8) + E(4)*E(4) = E(72).

Similarly,

Pref(E(http://www.cadsoftusa.com/)) = E(4)*E(7) + E(2)*E(4) = E(36).

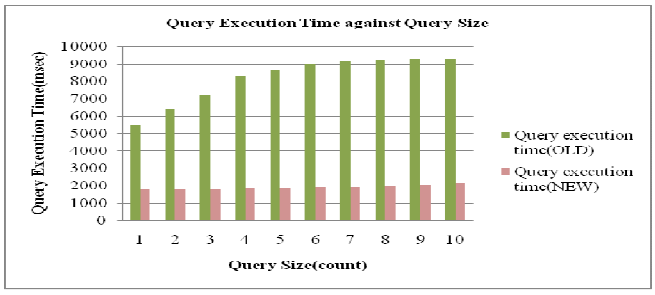

Fig. 8 Comparison of Query Execution Time against Query Size.

Table 4: Comparison of Query Execution Time against Query Size

Query size (count)	**Query execution time(OLD) (msec)**	**Query execution time(NEW) (msec)**
1	5444	1778
2	6380	1789
3	7211	1805

4	8267	1813
5	8647	1857
6	9023	1896
7	9145	1934
8	9234	1978
9	9289	2045
10	9315	2167

Then, the server forward these urls in descending order to the online profiler. At the server side, this calculation is possible only when we use Homomorphic encryption.

Encryption and decryption: The crawled data are encrypted and stored on server. Here user's request is also in encrypted form. The details about Fully Homomorphic Encryption over Integers are described in section 3. This type of encryption is applied only to the integer form of input data. In our case, the input data are URLs, words, and ranks. So, we need to convert these string inputs into integer form. For that here we use substitution technique before performing Homomorphic encryption. For substitution, take each input string and group them into four characters each, and then find the corresponding ASCII value. During decryption, initially perform Homomorphic decryption, then the reverse process of substitution method will happen.

6. Experimental Results

In this section, we show the experimental results of UPES. We conduct two experiments on UPES.

Experiment 1: Request size against Query size

Here we check request size against query size. In existing system, user passes a generalized profile when submitting a query. But in UPES, it passes only top five terms related to the submitted query. Fig.4 shows the user request to the server in existing system when a query "eagle" submitted in the search interface. It contains around 400 topics and server need to consider all these topics for personalized search results.The request size gradually increases when the number of topics in query increases. Fig.5 shows the request from user to the server and it contains only top five topics related to the submitted query. In this example, the submitted query is "eagle", and then the request contains only six topics and their corresponding ranks in the user profile. So, in the case of UPES, request size never exceeds query size plus five. Table.3 shows the resultant request size of the two techniques while varying the query size. Fig.7 illustrates the results of request size of both techniques by varying the number of topics in queries (from 1 to 10).

Experiment 2: Query execution time against Query time

Now we look at the query execution time of proposed system. Due to the bulk amount of request size, the query execution time of existing system is more than the proposed system. In existing system, web server need to consider all topics in generalized user profile for personalized search results. So it is a time consuming process. But in UPES, web server only take diminutive time for giving personalized search results to the user, because it only consider top five terms from user profile for submitted query. The following Table.4 shows the resultant query execution time of the two techniques while varying the query size. Based on these details, Fig.8 illustrates the results of query execution time of both techniques by varying the number of topics in queries (from 1 to 10).

7. Conclusion and Future Works

This thesis work provides a client-side and server-side privacy protection framework called UPES for personalized web search. In UPES, the server must automatically protect the users' privacy without customizing privacy requirements by the user. Because, here we applies Homomorphic encryption when request sent to the server. So there is no problem occurs when an eavesdropper gets this request. Since the encrypted data are stored at server, we can protect web server from all types of web attacks like URL manipulation attacks, trial and error attacks, etc. Besides this encryption, we also provide an authentication for server to protect it from attacks. Here, users' neither registers their personal information's nor customizes their privacy requirements. Thus using this framework, we can completely protect client-side and server-side privacy. The experimental results show that this technique functioned in the best possible manner with the least waste of time and effort. We are planning to develop a new Homomorphic encryption that we can directly apply in string type data by replacing the existing FHEI technique. Thus we can improve the performance and can extremely reduce the query execution time of framework. Also we have a plan to do multi profiling and re-ranking by using parallel computing.

References

[1] R.S.Bhadoria, D.Sain, and R.Moriwal, "Data Mining Algorithms for personalizing user's profiles on Web", Vol.1,issue 2 IJCTEE.

[2] A.Pretschner and S.Gauch,"Ontology-Based Personalizaed search and Browsing," Proc. IEEE 11th Int'l Conf.Tools with Artificial Intelligence,1999.

[3] L. Fitzpatrick and M.Dent,"Automatic Feedback Using Past Queries:Social Searching?," Proc.20th Conf.,1997.

[4] K.Sugiyama, K. Hatano, and M. Yoshikawa,"Adaptive Web Search Based on User Profile Constructed without any Effort from Users," Proc.13th Int'l Conf. World Wide Web (WWW), 2004

[5] M.Spertta and S.Gach,"Personalizing Search Based on User Search Histories,"Proc.IEEE/WIC/ACM Int'l Conf.Web Intelligence,2005.

[6] Y.Xu,K.Wang,B.Zhang and Z.Chen,"Privacy-Enhancing Personalized Web search,"Proc.16th Int'l Conf.,2007.

[7] L.Shou,H. Bai,K.Chen, and G. Chen,"Supporting Privacy Protection in Personalized Web Search",vol.26,February 2014.

[8] J. Yu, P. Lu, Y. Zhu, G. Xue and M. Li, "Toward Secure Multikeyword Top-k Retrievel over Encrypted Cloud Data", IEEE transactions on dependable and secure computing, vol. 10, no. 4, July/august 2013.

[9] S. Latha, M. Rajaram, and S. N. Sivanandam, "A Survey on Semantic Web Mining based Web Search Engines", VOL. 6, NO. 10, OCTOBER 2011 ARPN Journal of Engineering and Applied Sciences.

[10] M. Rami Ghorab, Dong Zhou, Alexander O'Connor, and Vincent Wade, "Personalised information retrieval: survey and classification".

[11] R.S.achake and Prof.G.P.Potdar "A Survey on Personalized Search: An Web Information Retrieval System".

[12] V.Kakulapati , Dr. D. Vasumathi and S.Jena "Survey on Web Search Results Personalization Techniques".

[13] X. Shen, B. Tan, and C. Zhai, "Implicit User Modeling for Personalized Search," Proc. 14th ACM Int'l Conf. Information and Knowledge Management (CIKM), 2005.

[14] Y. Zhu, L. Xiong, and C. Verdery, "Anonymizing User Profiles for Personalized Web Search," Proc. 19th Int'l Conf. World Wide Web (WWW), pp. 1225-1226, 2010.