CAPITAL UNIVERSITY OF SCIENCE AND
TECHNOLOGY, ISLAMABAD

ObScure Logging: A Framework to Protect and Evaluate the Web Search Privacy

by

Mohib Ullah

A thesis submitted in partial fulfillment for the
degree of Doctor of Philosophy

in the

Faculty of Computing
Department of Computer Science

2020

ObScure Logging: A Framework to Protect and Evaluate the Web Search Privacy

By

Mohib Ullah

(PC133004)

Dr. Sung Wook Baik, Professor

Sejong University, Seoul, South Korea

(Foreign Evaluator 1)

Dr. M. Akhtar Ali, Senior Lecturer

Northumbria University, Newcastle, UK

(Foreign Evaluator 2)

Dr. Muhammad Arshad Islam

(Thesis Supervisor)

Dr. Nayyer Masood

(Head, Department of Computer Science)

Dr. Muhammad Abdul Qadir

(Dean, Faculty of Computing)

DEPARTMENT OF COMPUTER SCIENCE

CAPITAL UNIVERSITY OF SCIENCE AND TECHNOLOGY

ISLAMABAD

2020

This work is dedicated to my parents, my siblings and my wife, without their prayers, support and encouragement this work would not have been completed. I must extend gratitude to my father (Haji Habibullah) and wife (MS. S Mohib) because if they had not been around, this work might not have been accomplished. This work is also dedicated to Muhammad Yusuf Khan, who is considered the real driving force behind this degree.

CAPITAL UNIVERSITY OF SCIENCE & TECHNOLOGY
ISLAMABAD

Expressway, Kahuta Road, Zone-V, Islamabad
Phone:+92-51-111-555-666 Fax: +92-51-4486705
Email: info@cust.edu.pk Website: https://www.cust.edu.pk

CERTIFICATE OF APPROVAL

This is to certify that the research work presented in the thesis, entitled "**ObScure Logging: A Framework to Protect and Evaluate the Web Search Privacy**" was conducted under the supervision of **Dr. Muhammad Arshad Islam**. No part of this thesis has been submitted anywhere else for any other degree. This thesis is submitted to the **Department of Computer Science, Capital University of Science and Technology** in partial fulfillment of the requirements for the degree of Doctor in Philosophy in the field of **Computer Science.** The open defence of the thesis was conducted on **August 13, 2020.**

Student Name : Mohib Ullah (PC133004)

The Examination Committee unanimously agrees to award PhD degree in the mentioned field.

Examination Committee :

(a) External Examiner 1: Dr. Kashif Munir,
Associate Professor
FAST-NUCES, Islamabad

(b) External Examiner 2: Dr. Azhar Mahmood,
Associate Professor
SZABIST, Islamabad

(c) Internal Examiner : Dr. Mohammad Masroor Ahmed
Assistant Professor
CUST, Islamabad

Supervisor Name : Dr. Muhammad Arshad Islam
Associate Professor
FAST-NUCES, Islamabad

Name of HoD : Dr. Nayyer Masood
Professor
CUST, Islamabad

Name of Dean : Dr. Muhammad Abdul Qadir
Professor
CUST, Islamabad

AUTHOR'S DECLARATION

I, **Mohib Ullah (Registration No. PC133004)**, hereby state that my PhD thesis titled, '**ObScure Logging: A Framework to Protect and Evaluate the Web Search Privacy** is my own work and has not been submitted previously by me for taking any degree from Capital University of Science and Technology, Islamabad or anywhere else in the country/ world.

At any time, if my statement is found to be incorrect even after my graduation, the University has the right to withdraw my PhD Degree.

(**Mohib Ullah**)

Dated: 13 August, 2020

Registration No : PC133004

PLAGIARISM UNDERTAKING

I solemnly declare that research work presented in the thesis titled "**ObScure Logging: A Framework to Protect and Evaluate the Web Search Privacy**" is solely my research work with no significant contribution from any other person. Small contribution/ help wherever taken has been duly acknowledged and that complete thesis has been written by me.

I understand the zero tolerance policy of the HEC and Capital University of Science and Technology towards plagiarism. Therefore, I as an author of the above titled thesis declare that no portion of my thesis has been plagiarized and any material used as reference is properly referred/ cited.

I undertake that if I am found guilty of any formal plagiarism in the above titled thesis even after award of PhD Degree, the University reserves the right to withdraw/ revoke my PhD degree and that HEC and the University have the right to publish my name on the HEC/ University Website on which names of students are placed who submitted plagiarized thesis.

(**Mohib Ullah**)

Dated: 13 August, 2020

Registration No : PC133004

List of Publications

It is certified that following publication(s) have been made out of the research work that has been carried out for this thesis:-

Journal Paper

1. **Ullah, Mohib**, Muhammad Arshad Islam, Rafiullah Khan, Muhammad Aleem, and Muhammad Azhar Iqbal. "ObScure Logging (OSLo): A Framework to Protect and Evaluate the Web Search Privacy in Health Care Domain." *Journal of Medical Imaging and Health Informatics* Vol. 9(6), pp. 1181-1190, 2019.

Conference Papers

1. **Ullah, Mohib**, Rafiullah Khan, and Muhammad Arshad Islam. "Poshida, a protocol for private information retrieval." *2016 Sixth International Conference on Innovative Computing Technology (INTECH)*. pp. 464-470. IEEE, 2016

2. **Ullah, Mohib**, Rafiullah Khan, and Muhammad Arshad Islam. "Poshida II, a Multi Group Distributed Peer to Peer Protocol for Private Web Search." 2016 *International Conference on Frontiers of Information Technology (FIT)*. pp. 75-80. IEEE, 2016.

Mohib Ullah
(Registration No. PC133004)

Acknowledgements

All praise to Allah, the most merciful and most magnificent. First, I would like to thank Dr. Muhammad Arshad Islam for the supervision of this dissertation. He was always available to guide during the entire research period. I would like to thank and acknowledge the unprecedented support of Mr. Rafiullah Khan, without his company, I would not be able to complete this degree. Special thanks to Dr. Abdul Qadir, Dr. Muhammad Aleem, and Dr. Nayyar Masood who were a source of inspiration for me. They were always available to boost my confidence and supported me in completing this degree. Mr. Muhammad Shahid and Dr. Asfandyar khan (Agriculture University Peshawar) have also done their part through providing their sincere help. This acknowledgment would be incomplete without mentioning the name of Dr. Inam Ul Haq (Khushal Khan Khattak University Karak) who extended his genuine concern and expertise to keep me on track. In the end, thanks to all the faculty and support staff of Capital University of Science and Technology, Islamabad.

Abstract

Web Search Engine (WSE) is an inevitable software system used by people around the world to retrieve data from the web. WSE stores search queries to build the user's profile and provides personalized results. These search queries hold identifiable information that can possibly compromise the privacy of the users. Preserving privacy in web search is the main concern of the users belonging to different walks of life. This research tries to highlight the loopholes imbedded in the available privacy preserving techniques. Besides, it aims at proposing some novel protocols with the least possible limitations. In this highly technological world, user is astonishingly surrounded by the amazingly advanced gadgets yet he is madly desirous to keep intact his privacy to the maximum. In order to preserve the Web search privacy of a user, this dissertation proposes a number of protocols such as a single group ObScure Logging (OSLo), a Multi Group ObScure Logging (MG-OSLo) and a Profile aware ObScure Logging (PaOSLo). This research work focuses on two main objectives. The first objective of this dissertation is to assess the local privacy and the profile privacy of a user through unlinkability and indistinguishability. The second objective of this dissertation is to evaluate the impact of group size, group count and the profile aware grouping on the local privacy and on the profile privacy of a user. Local privacy of proposed protocols has been evaluated by using probabilistic advantage being a curious entity and having linking query with the user. The profile privacy calculates the level of profile obfuscation using a privacy metric Profile Exposure Level (PEL). Computing the profile privacy of a user, a test has been performed over the same subset of AOL query log for two situations i.e. first, when the self-query submission is allowed and second, when self-query submission is not allowed. The privacy achieved by the proposed protocols has been compared with the state-of-the-art privacy-preserving protocol UUP(e) and co-utile protocol.

In the first protocol (OSLo), random users are grouped together to compute the impact of group size on the privacy of users in a single group design. In MG-OSLo, users are grouped by using non-overlapping group design and overlapping group design to measure the impact of group size and group count on the privacy of a

user. The calculation depicts that the probability of linking query with the user depends on the group size and group count i.e. larger the group size or higher the group count lower the probability of linking query with the user. Whereas users, having dissimilar interest, are grouped together in PaOSLO, in order to evaluate the impact of profile aware grouping on the privacy. The results show that OSLo provides 9.37% better privacy as compared to the co-utile and 6.67% better privacy than UUP(e). The multi-group has a positive impact on the local privacy and on the profile privacy of a user. The MG-OSLo preserves 19.9% better privacy as compared to co-utile and 9.1% better as compared to UUP(e). Similarly, The profile aware grouping (PaOSLo) further improves the profile privacy as compared to UUP(e) and OSLo. The PaOSLo has 10% less PEL as compared to UUP(e) and 2.5% less as compared to OSLo when it is simulated on the same dataset.

Contents

Author's Declaration v

Plagiarism Undertaking vi

List of Publications vii

Acknowledgements viii

Abstract ix

List of Figures xiv

List of Tables xvi

Abbreviations xvii

Symbols xix

1 Introduction 1

1.1 Motivation . . . 1
1.2 Standalone Methods . . . 5
1.3 Third-party Infrastructure . . . 5
1.4 Hybrid Technique . . . 6
1.5 Query Scrambling . . . 6
1.6 Distributed Schemes . . . 7
1.7 Objectives and Significance . . . 8
1.8 Research Question / Problem statement . . . 8
1.9 Contribution . . . 10
1.10 Dissertation Organization . . . 11

2 Literature Review 13

2.1 Standalone Schemes . . . 13
2.1.1 TrackMeNot . . . 14
2.1.2 GooPIR . . . 15
2.1.3 Dissociating Privacy Agent (DisPA) . . . 16
2.2 Third-party Infrastructure . . . 16
2.2.1 Scroogle . . . 16

2.2.2 TOR (The Onion Routing) . . . 17
2.2.3 Privacy-Preserving Framework using DLT and TOR . . . 17
2.3 Query Scrambling . . . 18
2.4 Hybrid Techniques . . . 18
2.4.1 Private Efficient and Accurate Web Search (PEAS) . . . 18
2.4.2 X-Search . . . 19
2.5 Distributed Schemes . . . 19
2.5.1 Indistinguishability Solutions . . . 20
2.5.2 Unlinkability Solutions . . . 27
2.5.3 Implementation of Distributed Protocols . . . 33
2.6 Summary of Distributed Protocols and Research Gap . . . 34
2.7 Privacy Evaluation Metrics . . . 36
2.7.1 Entropy . . . 36
2.7.2 Degree of Anonymity . . . 37
2.7.3 Profile Exposure Level (PEL) . . . 37

3 ObScure Logging (OSLo). 41
3.1 Introduction . . . 41
3.2 ObScure Logging (OSLo) . . . 44
3.3 OSLo Execution Process . . . 45
3.3.1 Connection Setup . . . 46
3.3.2 SQFC Selection . . . 46
3.3.3 Query Sending Process . . . 48
3.3.4 Query Shuffling . . . 49
3.3.5 Query Sending to WSE and Result Retrieval . . . 49
3.3.6 Result Decryption Process . . . 52
3.4 Dataset . . . 53
3.4.1 Dataset 1 . . . 54
3.4.2 Dataset 2 . . . 55
3.5 Privacy Mechanism . . . 56
3.5.1 Adversary Model . . . 56
3.5.2 Mechanism to Achieve Local Privacy . . . 57
3.5.3 Mechanism to Achieve Profile Privacy . . . 60
3.6 Privacy Evaluation . . . 62
3.6.1 Local Privacy Evaluation . . . 64
3.7 Results and Discussion . . . 67
3.7.1 Profile Privacy Evaluation . . . 68
3.7.2 Time delay of OSLo . . . 80
3.7.3 Performance Comparison of UUP(e) vs. OSLo . . . 83
3.8 Limitation of OSLo . . . 86
3.9 Conclusion . . . 87

4 Multi-Group ObScure Logging (MG-OSLo) 90
4.1 Multi-Group ObScure Logging (MG-OSLo) . . . 92
4.1.1 Entities . . . 92
4.1.2 MG-OSLo Execution Process . . . 93

4.2 Privacy Evaluation of MG-OSLo . . . 101
4.2.1 Local Privacy . . . 102
4.2.2 Profile Privacy . . . 110
4.3 Results and Discussion . . . 111
4.3.1 Profile Privacy of MG-OSLo: Self-Query Submission not Allowed Dataset 1 . . . 111
4.3.2 Profile Privacy of MG-OSLo: Self-Query Submission Allowed Dataset 1 . . . 113
4.3.3 Profile Privacy of MG-OSLo: Self-Query Submission Allowed Dataset 2 . . . 113
4.3.4 Profile Privacy of MG-OSLo: Self-Query Submission not Allowed Dataset 2 . . . 113
4.3.5 MG-OSLo VS OSLo VS Co-utile: Self-Query Submission Allowed Dataset 1 . . . 114
4.3.6 MG-OSLo VS OSLo VS Co-utile: Self-Query Submission Allowed for Dataset 2 . . . 117
4.3.7 MG-OSLo vs UUP(e) and OSLo: Self-Query Submission not Allowed Dataset 1 . . . 118
4.3.8 MG-OSLO VS UUP(e) and OSLo: Self-Query Submission not Allowed at Dataset 2 . . . 120
4.3.9 Time Complexity of MG-OSLo . . . 121
4.4 Conclusion . . . 122

5 Profile aware ObScure Logging (PaOSLo) 125
5.1 PaOSLo Description . . . 126
5.1.1 User Profile Construction . . . 127
5.1.2 Measuring Similarity between the User Profiles . . . 128
5.1.3 Cosine Similarity . . . 129
5.1.4 Profile Clustering . . . 130
5.2 PaOSLo Execution Process . . . 133
5.3 Privacy of PaOSLo . . . 136
5.3.1 Local Privacy of PaOSLo . . . 136
5.3.2 Profile Privacy of PaOSLo . . . 138
5.3.3 Profile Privacy Comparison of UUP(e), OSLo and PaOSLo . 139
5.4 Conclusion . . . 140

6 Conclusion and Future Work 142
6.1 ObScure Logging (OSLo) . . . 143
6.2 Multi Group ObScure Logging (MG-OSLo) . . . 144
6.3 Profile Aware ObScure Logging (PaOSLo) . . . 146
6.4 Limitation of Proposed Work . . . 147
6.5 Future Work . . . 148

Bibliography 149

List of Figures

1.1 Taxonomy of private web search 6

2.1 TrackMeNot [48] 14
2.2 DisPA: disassociation enactment demonstration [51] 15
2.3 The Onion Routing (TOR) model [33] 17
2.4 A path between a user and Web Server in a Crowds [38]. 21
2.5 Degree of Anonymity [38]. 22
2.6 Timeline of distributed privacy-perserving protocol 32

3.1 Activity diagram of user connection and SQFC selection 47
3.2 Activity diagram of query sending and result retrieval process . . . 48
3.3 Graphical representation of query sending process 52
3.4 Statistics of AOL query log by Peddinti and Saxena [49] 54
3.5 Probability of Head, After Tossing 59
3.6 ODP hierarchy of categories [83] 63
3.7 Probability of Head, after number of tossing 67
3.8 Average PEL of OSLo with Dataset 1 70
3.9 Avearage PEL of OSLo with Dataset 2 72
3.10 Avearage PEL of OSLo VS Co-utile with Dataset 1 74
3.11 Avearage PEL of OSLo VS Co-utile with Dataset 2 75
3.12 Avearage PEL of OSLo VS. UUP(e) for Dataset 1 78
3.13 Avearage PEL of OSLo VS. UUP(e) for Dataset 2 79
3.14 Time required to create a group 81
3.15 Time required to send a query and retrieve results 83
3.16 Delay comparison of OSLo vs other protocols 84
3.17 Number of groups required to simulate dataset 2 85

4.1 MG-OSLo: Activity diagram of query sending process 97
4.2 MG-OSLo: Graphical representation of query sending process . . . 98
4.3 Average PEL of MG-OSLo: self-query submission not allowed with dataset 1 112
4.4 Average PEL of MG-OSLo: self-query submission allowed Dataset1 112
4.5 Profile Privacy of MG-OSLo: Self-query submission allowed Dataset 2 114
4.6 Average PEL of MG-OSLo: self-query submission not allowed with Dataset 2 115
4.7 Average PEL of MG-OSLo VS. OSLo VS. Co-utile at Degree 1 and Degree 2 of ODP hierarchy for Dataset 1 115

4.8 Average PEL of MG-OSLo VS. OSLo VS. Co-utile at Degree 3 and Degree 4 of ODP hierarchy for Dataset 1 . . . 116
4.9 Average PEL of MG-OSLo VS OSLo VS Co-utile at Degree 1 and Degree 2 of ODP hierarchy for Dataset 2 . . . 117
4.10 Average PEL of MG-OSLo VS. OSLo VS. Co-utile at Degree 3 and Degree 4 of ODP hierarchy for Dataset 2 . . . 118
4.11 Average PEL of MG-OSLo VS. UUP(e) and OSLo at Degree 1 and Degree 2 for Dataset 1 . . . 119
4.12 Average PEL of MG-OSLo VS UUP(e) VS OSLo at Degree 3 and Degree 4 for Dataset 1 . . . 120
4.13 Average PEL of MG-OSLo VS. UUP(e) VS. OSLo at Degree 1 and Degree 2 for Dataset 2 . . . 121
4.14 Average PEL of MG-OSLo VS. UUP(e) VS. OSLo at Degree 3 and Degree 4 for Dataset 2 . . . 122

5.1 PaOSLO: Activity diagram of User profile construction and clustering 127
5.2 Categories at first degree of the ODP hierarchy [83] . . . 128
5.3 Sample ARFF file . . . 132
5.4 Term count in three cluster . . . 133
5.5 Term count in four cluster . . . 134
5.6 Term count in five cluster . . . 134
5.7 PaOSLo: Acitivity diagram of Group creation and SQFC selection process . . . 135
5.8 PaOSLo: Activity diagram of query sending and result broadcasting process . . . 136
5.9 Average PEL of UUP(e) VS. OSLo VS. PaOSLo . . . 140

List of Tables

2.1 Simulators used for the implementation of distributed protocol . . . 34
2.2 Summary of distributed protocol . . . 35

3.1 AOL query log attributes and description [76] . . . 53
3.2 Dataset 1: Range of queries sent by a user . . . 55
3.3 Dataset 2: Range of queries sent by a user . . . 55
3.4 Distributed protocol shuffling method . . . 59
3.5 Example of query categorization by ODP [82] . . . 63
3.6 Profile of a user X at different degrees . . . 63
3.7 OSLo average PEL self-query submission allowed, dataset 1 . . . 70
3.8 OSLo average PEL self-query submission allowed dataset 2 . . . 71
3.9 Co-utile protocol: Average PEL, self-query submission allowed for dataset 1 . . . 73
3.10 Co-utile protocol: Average PEL, self-query submission allowed for dataset 2 . . . 73
3.11 Average PEL of OSLo VS. UUP(e) self query-submission not allowed dataset 1 . . . 75
3.12 Average PEL of OSLo VS. UUP(e), self query-submission not allowed for Dataset 2 . . . 77
3.13 Simulation details under controlled environment: equipments . . . 81
3.14 Simulation parameters . . . 84

5.1 Query classification of queries by ODP into a hierarchy of categories. [82] . . . 129
5.2 Query categorization by ODP of a sample user “3978802” of AOL query log . . . 130
5.3 Terms extracted for degree 1 of user “3978802” . . . 130
5.4 Terms extracted for degree 1 of user “280617” . . . 131
5.5 Similarity between sample seven users’ profile at degree 1 of ODP hierarchy . . . 131
5.6 Number of users in each cluster after K-Mean clustering . . . 131
5.7 Average PEL comparison of UUP(e), OSLo and PaOSLo . . . 139

Abbreviations

AnonID	Anonymous Identifier
AES	Advance Encryption Standard
AOL	American Online
CS	Core Server
DisPA	Dissociating Privacy Agent
SQFC	Search Query Forwarding Client
GSQFC	Group Search Query Forwarding Client
OSLo	ObScure Logging
MG-OSLo	Multi Group ObScure Logging
PaOSLo	Profile aware ObScure Logging
ODP	Open Directory Project
NLP	Natural Language Processing
HTTP	HyperText Transfer Protocol
TOR	The Onion Routing
PEAS	Private Efficient and Accurate Web Search
PIR	Private Information Retrieval
DB	Database
pf	Probability of Forwarding
eMsg	Encrypted Message
eQ	Encrypted Query
eQ_Msg	Encrypted Query Message
eAnsMsg	Encrypted Answer Message
OAS	Optimized Arbitrary Size
P2P	Peer-to-Peer
PEL	Profile Exposure Level

UUP	Useless User Profile Protocol
UUP(e)	UUP extended
UPIR	User Private Information Retrieval
WSE	Web Search Engine
WWW	World Wide Web

Symbols

H	Entropy
H_{Max}	Maximum Entropy
I	Mutual Information
pr	Probability
q	query
r	result

Chapter 1

Introduction

"Every man should know that his conversation, his correspondence and his personal life are private".

Lyndon B Johnson, President of United States (1963-69)

1.1 Motivation

Being an enormous warehouse of data, the World Wide Web (WWW) is a storehouse to a range of documents including text, images, video, audio, etc. Today, data is generated at great speed from a variety of fields. Information about anything and everything is uploaded on the WWW. We depend on WSEs to search for specific topics or information on Web. The WSE has become an integral part of daily life of Internet users. People from all over the world and from all walks of life, search for pertinent information available on the Internet. WSEs are browsed everyday by billions of users to search for different information by entering few words called queries. In 2002, 52% Americans used the WSEs, and this percentage rose to 73% in 2012 [1]. The current Internet live statistics show that Google answers around 72 thousand queries in one second [2]. Research shows that online users are more satisfied with the performance of search engines than ever [3, 4]. However, they are very concerned about their privacy because of the personalized search results and advertisements that appear as a result of their search [1]. The personalized search result is more appealing to users as it matches their interest.

A typical query is almost three words long revealing less information about the actual interest of the user. Additionally, sometimes certain words can cause ambiguity. Consider a search query on "mouse"; it is an animal and a computer device. Similarly, another search query "apple", it is a fruit and a technology company that sells electronic devices. In such cases, the WSEs use the user's profile to show relevant results [5, 6]. Research shows that WSEs evaluate the query log through certain algorithms to profile the users. Moreover, WSEs infer the interests of the user to provide personalized search results [7, 8]. In order to provide relevant results, WSE builds a user profile considering their interest, profession, preferences and previous searches. According to the authors [9], the user's profile improves search results. For an accurate response, the user's privacy is at risk. However, for absolute privacy, the user has to compromise on the accuracy of search result. While the user's profile is an asset for WSEs, it reveals sensitive data about the user [10]. For example, the user's profile supports retrieving the related materials from the Internet, while simultaneously it threatens the user's privacy as well. Most often a user's queries contain important information like Unique User ID, name, user's employer's details, location, etc. Moreover, a query may enclose health information, gender orientation, religion, politics, faith, believes, etc., which can be exceptionally sensitive data for the possessor [11].

According to studies, users sometimes query their own name, social security number, or other people's information [12]. WSEs are able to provide personalized results because the WSEs record all the submitted queries in a query log. A typical query log may contain user-submitted query contents, the machine IP address, operating system details, browser type, the query's date & time, browser language, preferences, and cookies that are possibly used to recognize the user [5, 9, 13, 14]. In the year 2010, a research study analyzed device fingerprinting over a data collected from 470,161 Internet users' browser setting who visited the Panopticlick [1] website concluded that browser setting parameters (screen dimensions, font, timezone, user agent strings, language settings, plugins) serve as a global identifier to uniquely identify a user anywhere on the Internet [15]. A user can block the cookies and disable an account login for web surfing, but there is no disable option

[1] https://panopticlick.eff.org. (accessed 25 June 2019)

for the device fingerprinting [16, 17]. Furthermore, disabling cookies can make a user more vulnerable to be identified because such settings make a user unique.

Likewise Nikiforakis et al. established that companies use a concealed flash object to determine a user who tries to hide his identity through a proxy server and link the queries to the user [18]. There are some non-explicit identifiers like zip codes, date of birth, gender, that if combined with the publicly available data they can be linked directly with the user's profile. The release of a query log poses serious risks in terms of privacy [19]. The disclosure of sensitive data to a third party (e.g., advertiser, media, etc.) [7, 12] can be exploited for business purposes or to retrieve information about competitors [19].

A survey [1] conducted in 2012 showed that 65% of users believed that it was unacceptable to record their searches, 73% of them were not happy with WSE keeping track of their data, and 68% of these surveyed individuals were not interested in advertisements they received due to profiling. The query log is a precious resource to the WSEs [20] since they analyze a person's query log to retrieve relevant search results. According to Cooper [9], the core privacy threat of storing a query log is the disclosure to advertising agencies and media. Additionally, the query log frequently holds sensitive data about users, and the dissemination of such data violates one's privacy [12]. The major privacy scandal is the release of the AOL log in 2006 [21, 22] where twenty million queries generated by 658000 users in three months of the time were published for research purpose. Before the search log was released, data was anonymized with the intention that no one should be able to link a query to the user, e.g.they replaced the IP address with a unique ID (pseudo-ID). However, this search log release had serious consequences, as data was not properly anonymized; a 62-year old lady named Thelma with ID 4417749 was successfully discovered [21–25]. The users of the WSE especially those who were using AOL were upset with the identification of the user "Thelma", and some users launched legal actions against AOL. A query log may also become a subject to a subpoena such as in an incident that happened in 2006 when the US Department of Justice (USDJ) issued a summon [26] to AOL, Yahoo, Google and Microsoft to provide the query log [27]. The issue was to find out if the Internet

filters were effectively protecting children from adult contents on the Internet as a part of the litigation of an Internet child safety law [9]. All search engines provided the information required by the court, however, Google gave only limited information [9, 28, 29]. Likewise, in some cases, a WSE compelled by the court to disclose individual queries involving a divorce or civil lawsuit as a part of the evidence [9]. Furthermore, in 2014, eighty million health records were lost by the US second largest insurance company [30, 31]. These issues triggered the movement of privacy preservation among members of the online community to eliminate the threat of malicious efforts to expose their identity. These incidents put a question mark on WSEs' policy about user privacy. Many users demanded that WSE should not maintain the query log [1]. Consider a situation where a company is interested to promote an employee Mr. X, but Mr. X is searching for a specific disease supposedly Hepatitis. If the company comes to know about the employee's search queries the company could suspect Mr. X as a Hepatitis patient and may drop the idea of promotion; as a worst case scenario they may terminate his job [8]. Another example is that if a company is searching for certain ideas or materials regarding a new product to be launched, competitors can find out about the company's plans based on the queries and can achieve a competitive advantage. Furthermore, suppose a young woman is searching for information about morning sickness or late cycle. The words she needs to use for the query will necessarily reflect that she is possibly pregnant. These examples provide the evidence that search queries include very personal data about a person that needs to be kept private. In 2009 Eric Schmidt CEO of Google, responded to privacy concerns that "if you have something and you want no one knows it then you should not be searching it in the first place to achieve a high level of privacy". For better results, all WSEs, including Google, keep information for some period [32]. Given the high number of cases in which user data is not sufficiently protected, there is a need for actions to protect their privacy and prevent WSEs from profiling them.

So far, many techniques have been proposed to protect the privacy of the users during their Web search. These techniques can be classified into five main groups, i.e., standalone methods, third-party infrastructure, hybrid technique, query scrambling and distributed techniques. The standalone methods and query scrambling

aim at achieving indistinguishability **(obfuscating the user profile maintained by the WSE)**, whereas, the third-party infrastructure and distributed technique aim for unlinkability **(hiding the identity of an individual)**. Figure 1.1 shows the well-known privacy preserving schemes; the Section below gives the description of each technique.

1.2 Standalone Methods

The Standalone methods protect the privacy of the user against the profiling of WSE through indistinguishability. The standalone methods do not hide the identity of a user but they obfuscate the profile of a user by sending the fake queries to the WSE[10]. Standalone schemes are also called single-party infrastructure. The popular standalone schemes are TrackMeNot, GooPIR and DisPA. The detail description and working of each scheme are presented in Section 2.1.

1.3 Third-party Infrastructure

Another approach to achieve the Web search privacy is to use third-party infrastructure. A number of third party infrastructures such as scroogle[2], anonymizer [3], and The Onion Routing TOR [33] are available. These infrastructures are capable of attaining the privacy of a user through unlinkability, i.e., hiding the identity of a user. The user, using third-party services to achieve privacy, has to trust those servers. The scroogle and TOR are the popular third-party infrastructures adopted to enforce unlinkability. The description of each of these techniques is given in Section 2.2. Internet search engines like DuckDuckgo [4], Ixquick [5], and Yippy[6], etc., which distinguish themselves from other search engines by not profiling their users and by showing all users the same search results for a given search term. However, the risk of privacy breach remains the same.

[2]http://scroogle.org/
[3]http://www.anonymizer.com/
[4]duckduckgo.com
[5]ixquick.com
[6]yippy.com

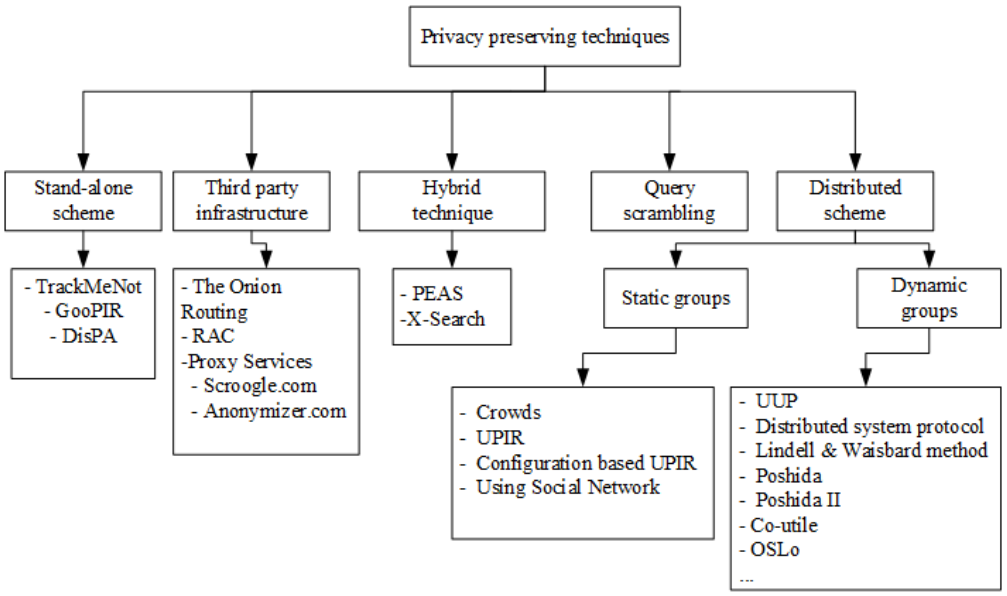

FIGURE 1.1: Taxonomy of private web search

1.4 Hybrid Technique

This approach adoptes the concept of both unlinkability and indistinguishability. The unlinkability is achieved by sending queries through third-party infrastructure like TOR, or proxy services etc and indistinguishability is attained by sending bogus queries with the real queries using RSS seed queries. The popular hybrid techniques are PEAS [34] and X-Search [35], the description of each hybrid technique is given in section 2.4.

1.5 Query Scrambling

Arampatzis et al. [23] proposed a technique for achieving the privacy of users which was termed as versatile query scrambling on the basis of their previous work [36]. This technique, instead of hiding the identity of a user, changed the user query in such a way that the WSE did not understand the aim of the users' interest. The technique was aimed at scrambling the query in such a way that the actual interest of the users remained hidden. The resultant scrambled query loosely corresponded to the actual interest and thus distorted the user's query meaning. The real query was divided into multiple scrambled queries and all those queries were then submitted to the WSE. The list of results of the scrambled queries

was collected and a scrambled ranking was applied on the list to get the actual interest of the user called descrambling. Query scrambling had low accuracy, as the scrambled query results hardly matched the initial query results.

1.6 Distributed Schemes

This technique works by the cooperation of multiple users with the intention of diffusing the identity of a user among the group users. For example, a group of users collaborates with each other to forward queries to the WSE. One member of the group forwards another user's queries and his/her query is thus forwarded by another user and vice versa. A kind of role changing takes place in the group for the sake of profile obfuscation. Many distributed techniques, which are proposed over the course of time, have been detailed in the literature review.

The key advantage of distributed schemes over the other privacy-preserving methods is that they achieve both unlinkability and indistinguishability. *Unlinkability is disassociating the query and user* i.e., unlinkability, is achieved if a query cannot be linked to the user. *Indistinguishability is obfuscating the user's profile by sending other users' queries* [37]. The privacy of the user is evaluated both locally relative to the group entities and to the WSE. This work defines two new terms i.e., local privacy and profile privacy.

Definition 1. Local Privacy: The privacy of the user relative to the peer entities (group users and core server) that cooperate in forwarding query to the WSE. Local privacy of the user is considered preserved, if the following two objectives are achieved. i). the user achieves unlinkability, i.e, no peer entity can link a query with the user, and ii) the contents of query and result of the query remain hidden from the group entities.

Definition 2. Profile Privacy: The privacy of the user relative to the WSE. Profile privacy is achieved through indistinguishability, i.e., a user obfuscates their profile by sending queries of other users and the WSE cannot build the accurate profile of a user.

1.7 Objectives and Significance

The importance of the Web search privacy is undeniable. Although there have been a great number of research efforts preserving the user's privacy in such a way that no entity can link the query to the originator and thus making WSE incapable of building a reliable profile has not been achieved. This research proposes a number of distributed privacy-preserving protocols that preserves the privacy of a user through unlinkability and profile obfuscation (indistinguishability). The primary objective of this dissertation is to propose a framework that preserves the Web search privacy of a user. The privacy shall be preserved in a way that a query can not be linked with the user and the WSE shall not build an accurate profile of a user.

To the best of our knowledge, the privacy of user executing distributed privacy-preserving protocol has never been evaluated in terms of unlinkability and indistinguishability. In this dissertation, the privacy of the user is evaluated from two dimensions, i.e., the local privacy and the profile privacy. The former is used to assess the unlinkability and the later is used to measure the indistinguishability. This work aims at helping both the institutions and the people of all categories ranging from academic institutions to national security and common persons who wish to remain anonymous while searching queries on the WSE. This research proposes a distributed protocols to preserve the privacy of a user in a situations when a group of users make a coalition to compromise the privacy of a user.

1.8 Research Question / Problem statement

Many distributed protocols have been proposed to preserve the privacy of a user by providing the unlinkability and indistinguishability. However, we have identified several limitations in the available distributed schemes [7, 38–45]:

1. The users occurring in a path between the query initiator and the WSE can see the query and the result, hence compromising the privacy of the user [38, 43, 44].

2. Users having access to the common memory location can see the query content, as the query is encrypted under the key K_l associated with memory location[40, 41].

3. The query results are broadcasted in unencrypted form, giving peers an idea of what is being searched inside the group [7, 40].

4. The peer agent (user) knows the exact query of another user, also, getting the query answer is another prime concern in co-utile protocol [45].

5. A user cannot submit his/her query to WSE, giving an attacker a clue to link the query with the originator.

6. Users are grouped randomly, making them susceptible to group up with those users who have similar interest, hence reducing the level of profile obfuscation.

7. The privacy of a user has never been evaluated for multiple groups.

To the best of our knowledge, the existing distributed protocols do not preserve and evaluate comprehensively the privacy of the user relative to the peer users involved in forwarding the query to the WSE (local privacy) and against the profile of WSE (profile privacy) while executing the distributed protocol. The privacy of the user must remain intact against the external attacks such as an eavesdropper, hacker, etc. Achieving the privacy of the user relative to the WSE, group peers, the central server, and an external attacker is the core objective of this research.

RQ 1. How to improve the local privacy and profile privacy of a user in a private web search?

RQ 1 (a). What will be the effect of allowing self-query submission and not allowing self-query submission on the profile privacy of the user?

RQ 1 (b). How does the size of the group and group count affect the privacy and performance of the protocol?

RQ2. What is the effect of random grouping and profile aware grouping on the privacy of the user?

1.9 Contribution

The contributions of this dissertation include:

1. A single group distributed privacy-preserving framework called ObScure Logging (OSLo) is proposed that reduces the limitations mentioned in existing schemes [7, 38–41, 46] and preserves the privacy of the user in a private web search.

2. The OSLo evaluates the local privacy (unlinkability) and profile privacy (indistinguishability) of a user by:

   (a) The formal analysis of local privacy by computing the probability of linking a query by a curious entity with the user.

   (b) The profile privacy is evaluated by computing the magnitude of profile obfuscation through OSLo using privacy metric Profile Exposure Level (PEL).

   (c) To compute the impact of group size on local privacy and profile privacy.

3. A Multi-Group distributed privacy-preserving protocol (MG-OSLo) is proposed that evaluates the impact of multiple groups on local privacy and profile privacy. Several formal design techniques are used to form groups, i.e., non-overlapping group design, overlapping group design, and balance incomplete block design (BIBD).

   (a) Details analytical analysis of local privacy of group design methods is performed. A MG-OSLo provides what probabilistic advantage an entity have while linking query with a user.

   (b) Detailed empirical evaluation is provided to measure the privacy of a user executing MG-OSLo relative to WSE. An experiment is performed to calculate the magnitude of profile obfuscation for multiple group count over a privacy metric PEL.

   (c) Impact of group size and group count on local privacy and profile privacy relative to the profiling of WSE is investigated.

4. Profile aware ObScure Logging (PaOSLo) is proposed that investigates the impact of grouping users having dissimilar interest on profile privacy of a user.

   (a) A cluster of users having similar profile interest is created using K-mean algorithm and to compute the similarity between the users' profile using cosine similarity measure.

   (b) A group of users by selecting each user from a different cluster is created.

   (c) An experiment is performed to compare the level of profile obfuscation using profile aware grouping and randomize grouping.

1.10 Dissertation Organization

The remaining portion of the dissertation is organized in such a way that chapter 2 presents the existing work, chapter 3 proposes OSLo and explains the evaluation of a user privacy, chapter 4 describes the MG-OSLo. The impact of profile aware grouping is explained in Chapter 5. The conclusion and recommendation for the future work are expounded in Chapter 6.
An overview of each chapter is detailed below.

Chapter 2- This chapter explains the existing privacy-preserving schemes and privacy evaluation parameters. The existing schemes include standalone alone scheme, third-party infrastructure, proxy services, hybrid schemes, and distributed schemes. Each of the existing schemes has further techniques and protocol, which are elaborated in the next chapter. There are different metrics that evaluate the privacy of a user, however, the entropy, degree of anonymity and profile exposure level (PEL) are detailed in this chapter.

Chapter 3- This chapter proposes a novel framework OSLo. This framework consists of two parts. The first part proposes a protocol which preserves the Web search privacy of a user by providing unlinkability and indistinguishability. The second part of the framework evaluates the local privacy and profile of a user. A

probabilistic model is used to evaluate the local privacy (unlinkability) by computing the probabilistic advantage a curious user has in linking a query with the originating users. The profile privacy (indistinguishability) measures the magnitude of profile obfuscation by sending the queries of the group user. The proposed privacy and performance of OSLo is compared with the state-of-the-art privacy-preserving protocols co-utile and UUP(e). The profile privacy is compared for two situations: first, self-query submission is allowed and second, self-query submission is not allowed.

Chapter 4- A multi-group distributed protocol MG-OSLo is proposed to investigate the impact of multiple groups on local privacy and profile privacy. Users are grouped using overlapping and non-overlapping group design approach. The profile privacy of a user is evaluated by changing the group count and group size. The profile privacy of single group (OSLo and UUP(e)) is compared with MG-OSLo.

Chapter 5- In the existing distributed privacy-preserving protocol groups are made on the users' first come first grouped basis. In this approach, a user may be grouped with those users having similar interest making the profile less obfuscated. This chapter proposes a novel profile aware ObScure Logging (PaOSLo) protocol that first clusters the users having similar interest, and then selects a user from each different cluster. The profile privacy of a user is compared with OSLo and UUP(e) to investigate the difference between random grouping and profile aware grouping.

Chapter 6- This chapter details the key findings of this research, identify the open challenges to the private web search and the future directions to enhance the web search privacy of a user.

Chapter 2

Literature Review

This chapter focuses on the detailed explanation of the available privacy preserving techniques. For instance, Steven Brier first highlighted the importance of web privacy, in his article "How to Keep Your Privacy: Battle Lines Get Clearer" published on January 13, 1997, in the New York Times. Brier mentioned whatever you do on the internet is recorded [47]. The ISP, marketing maven, and many other people are keeping track of your work. The privacy and anonymity of your internet browsing are hardly guaranteed.

As mentioned in Section 1.1, privacy-preserving techniques can be classified into five main categories i.e., standalone scheme, third-party infrastructure, hybrid approach, query scrambling, and distributed schemes.The details of each of these is as follows:

2.1 Standalone Schemes

Standalone methods also called single party system, try to achieve indistinguishability. These schemes are usually available as a browser plugin. A scheme in this category sends bogus queries with original queries to mask the user profile among the bogus/fictitious queries. Standalone schemes do not suffer any network delay is the main advantage of standalone schemes. Some of the famous standalone techniques are mentioned below.

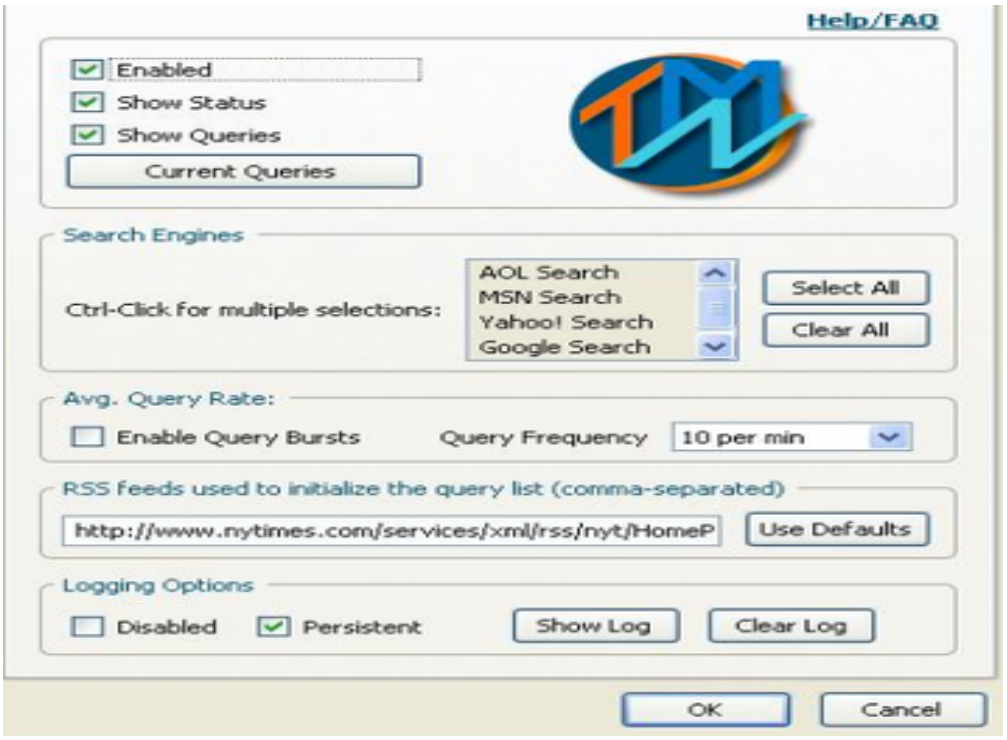

FIGURE 2.1: TrackMeNot [48]

2.1.1 TrackMeNot

Nissenbaum and Howe [48] proposed a technique as a Internet browser plugin to programmatically create a query seed file and send some noise queries with the original query. This approach hides the user's original query among the noise queries, and thus obfuscates the user's profile. However, this puts some extra communication overhead of sending false queries to the WSE. Furthermore, the automatically generated queries may sometime be more dangerous than the original query and may put the user in trouble. TrackMeNot queries are machine generated and hence easily distinguishable from the human generate queries since the peripheral means used for producing false queries, such as dictionaries or RSS feeds, make it possible for the WSE to distinguish between the false queries and the real query [49]. Figure 2.1 shows the browser extension of TrackMeNot, a user has the option to select multiple search engines. Additionally, the user is required to give the query list of the RSS feed. Since August 2006, when the initial version of TMN was made publicly available free of charge, in the year 2009 the statistics states that there have been over 350K downloads [48]. However, some critiques on TrackMeNot maintain that no matter how many false queries are generated to

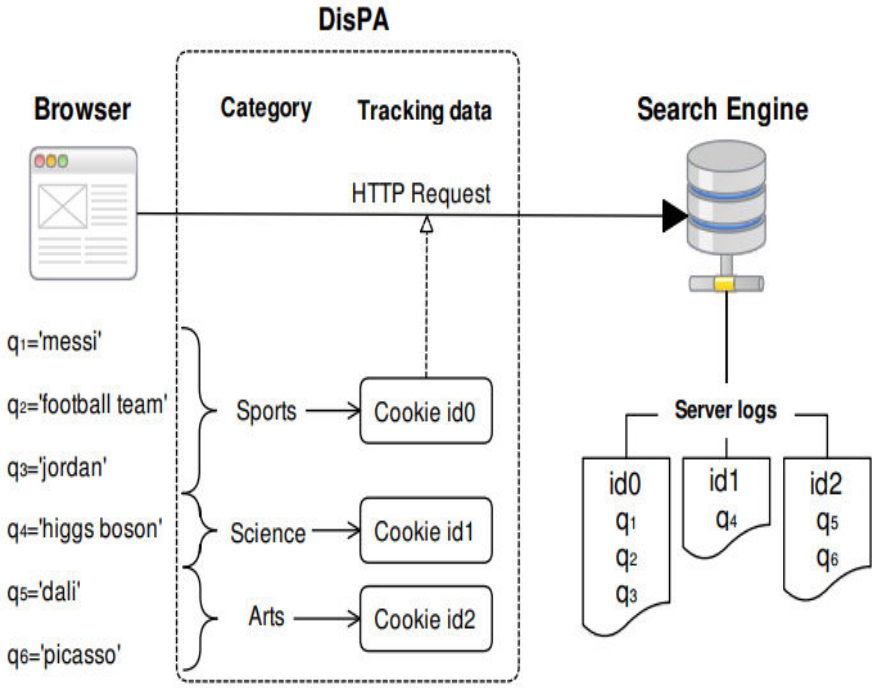

FIGURE 2.2: DisPA: disassociation enactment demonstration [51]

hide the original query, if they want to track the user they will certainly find his or her original interest.

2.1.2 GooPIR

Domingo-Ferrer, Solanas and Castell'a-Roca [50] proposed a GooPIR system to disguise the user's query before sending to WSE. It breaks the query into several keywords then masks the keywords with k-1 false keywords that have the same frequency as the original keywords. These masking keywords are chosen so that they have a frequency similar to the target keywords. The GooPIR system did not achieve popularity due to its poor performance, because the false queries and phrases that used to appear in the query were already known. The queries sent through GooPIR were distinguishable through machine learning attacks. GooPIR was also vulnerable to the de-anonymization attack. Petit et al, made a similarity attack and concluded that if the WSE has already a profile of the user, GooPIR was not able to effectively protect the privacy of the user and his or her actual interest [34].

2.1.3 Dissociating Privacy Agent (DisPA)

DisPA was proposed to provide privacy to the user relative to the WSE by hiding the identity of a user using multiple HTTP cookies [51]. DisPA assumed that WSE builds multiple profiles for a person based on multiple cookies. Figure 2.2 shows the disassociation enactment. The DisPA intercepts the user's HTTP request to the WSE and the intercepted query is classified into different categories. The classified query by the DisPA is made to appear as a separate request to the WSE from a different user. The aim of this approach is to reduce the disclosure of information and thus sufficient protection of users' privacy cannot be guaranteed.

2.2 Third-party Infrastructure

The third-party infrastructures are used to obtain the unlinkability. The Onion Routing (TOR) [33], Proxy services like Scroogle [1] and anonymizer [2] are the popular approaches to achieve web search privacy. Additionally, there are many websites like DuckDuckgo [52], Ixquick [53], and Yippy [3], etc., which claim to provide privacy by not profiling the users since they never store any user's personal information. If any user performs a search on those websites, the user will get the same search results because they do not store the user profile. However, the risk of privacy breach remains the same as like a WSE.

2.2.1 Scroogle

It was launched in 2003 and went offline in 2012. By then it was answering 350K queries per day, Scroogle [4] was forwarding queries to Google on behalf of the user. It was popular among searchers who wanted to get Google search results in a private setting. Being a proxy service just like anonymizer [5] or ZenMate [6], yet Scroogle started profiling in the same way as was done by the WSE.

[1] http://scroogle.org/
[2] http://www.anonymizer.com/
[3] www.yippy.com
[4] http://scroogle.org/
[5] http://www.anonymizer.com/
[6] https://zenmate.com/ last accessed Feb 23, 2020

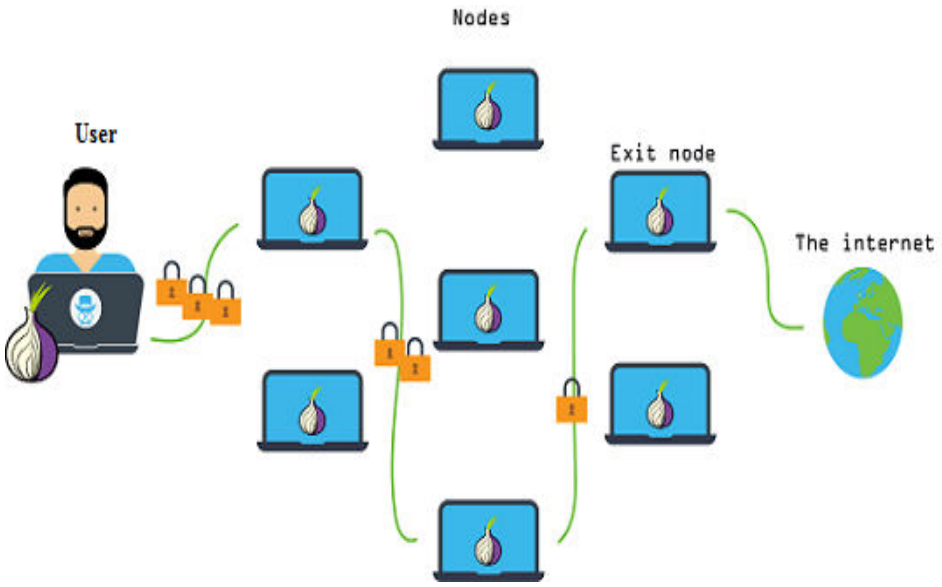

FIGURE 2.3: The Onion Routing (TOR) model [33]

2.2.2 TOR (The Onion Routing)

TOR is the group of volunteer-operated network servers to provide privacy [33]; it actually consists of multiple proxies and is used to hide the identity (IP address) of an individual on the internet. It was not specifically designed for Web anonymity. Although TOR provides the anonymity at the network layer, the WSE can identify a person at the application layer. With TOR a user gets a very slow response and none-personalized results. Figure 2.3 shows how a user establishes a path to WSE through multiple proxies.

2.2.3 Privacy-Preserving Framework using DLT and TOR

Raza et al, proposed a framework for privacy-preserving, distributed search engines using the topology of DLT and Onion Routing [54]. They reported that since the search results are heavily dependent on WSEs run by central authorities, the WSEs and central authorities not only compromise the privacy of an individual by keeping track of his or her daily searches but also provide biased results for the queries. To ensure privacy, Raza et al, used distributed ledger technology using TOR to search for the web contents and ensure anonymity. Their proposed framework consisted of four types of participant i.e. user, Tor nodes, search nodes, and Tor block nodes with the assumption that the majority of the participants

were non-malicious. The privacy of the proposed framework was mainly dependent on the TOR layer, whereas performance relied on the number of nodes in the TOR. As discussed in section 2.2.2, although TOR can provide network-layer anonymity however WSEs can identify a user through device fingerprinting.

2.3 Query Scrambling

The query scrambling technique conceals the profile of a user by not sending the actual query directly to the WSE. Instead, the user query is broken into multiple terms, whereas, each query term is sent separately to the WSE. The result for each query term is collected and descrambled to get an answer for the original query. However, query scrambling had two major shortcomings. First, the result retrieved had poor quality, it loosely corresponded to the actual interest of the user and second, query scrambling never tried to hide the identity of a user [13, 24].

2.4 Hybrid Techniques

Hybrid techniques are used to achieve both unlinkability and indistinguishability. This technique sends a list of fake queries with the original queries through a third-party infrastructure. In such case the profile of the user is obfuscated with the fake queries. As the queries are sent through the third party infrastructure like TOR or a proxy server hence the user achieves unlinkablity. PEAS and X-Search are the two most popular hybrid techniques to achieve Web search privacy, through unlinkability and indistinguishability.

2.4.1 Private Efficient and Accurate Web Search (PEAS)

Petit et al. [34] proposed PEAS to provide indistinguishability and unlinkability to a user in private web search. PEAS achieves indistinguishability by adding K fictitious queries with the original query using a logical OR operator. Whereas, the unlinkability is attained by sending the set of queries through a proxy server.

The results show that PEAS decreased 81.9% linking of queries with the originator user as compared to the GooPIR. The unlinkability is proved through the machine learning attack. However, PEAS used a very weak adversary model of non-collaborating proxies[35].

2.4.2 X-Search

X-search is an alternative approach of PEAS to provide indistinguishability and unlinkability [35]. X-Search has three entities i.e., a client, a proxy node (a trusted software Guard Extension install on a proxy node) deployed over the untrusted cloud and a WSE. The indistinguishability preserves the privacy by sending bogus queries which are generated from users previous search history and unlinkability between the client and WSE is achieved through the proxy node. However, Pires et al.[55], identified that X-Search and PEAS are not convincing techniques as they become simply blocked by search engines that have aggressive anti-boot strategies.

2.5 Distributed Schemes

Distributed schemes work with the cooperation of multiple users, every user forwards another user's query and vice versa. Distributed schemes achieve both unlinkability and indistinguishability to preserve privacy. The profile of a user is obfuscated with the real queries of other users in order to attain indistinguishability. Hence, the risk of a query being classified as the machine-generated query does not exist with distributed schemes. Accomplishing indistinguishability through the real queries of real users is the major advantage of the distributed scheme over other schemes(standalone, hybrid schemes). A user of distributed schemes achieve unlinkability when his or her query is forwarded by another user, hence, the WSE will not be able to link a query with the user directly. There are many distributed schemes proposed in the recent times, some techniques focused on indistinguishability alone while others focused on both unlinkability and indistinguishability. The sections below detail the temporal advances made in distributed schemes to achieve the Web search privacy.

2.5.1 Indistinguishability Solutions

Private Information Retrieval (PIR) is a technique proposed to retrieve an element from the database with the aim of not allowing the database to find out about the interests of the user. In order to make the database unable to get the actual interests of the user, Chor et al. [56] proposed PIR with the supposition that more than one copies of the database might be stored at different locations and these copies should not communicate with each other. For obtaining an item X_i privately from the database, Chor et al. described a technique in which the user was supposed to send queries to the copies of the same database, i.e., set S $\subseteq$ [n] (i.e., each index j $\in$ [n] was selected with probability 1/2). For instance, the user was supposed to send S to DB1 and S $\oplus$ i to DB2. By applying the exclusive OR operation on the results of the queries, the user may retrieve the item Xi successfully without letting the databases know his actual interests. In the light of a single database, Chor et al. assumed that a complete copy of the database may be considered in order to obtain the required interests while keeping intact the privacy. Later in 1997, Kushilevitz and Ostrovsky [57] came up with the idea that PIR could be applied on a single database by using algebraic properties and methods of Goldwasser–Micali public-key encryption.

In 1998, Reiter and Rubin [38] stressed the fact that encryption only protects the contents of a user's data. However, eavesdroppers can still capture the user's IP address, the duration of communication, and the partner. A WSE knows the client's IP address, his/her query and by digging other patterns from it, a WSE can invade the privacy of a client. Reiter and Rubin introduced a technique called Crowds for achieving anonymity in a web transaction. To retrieve data privately from WSE, a user is first required to join the crowd before making a web transaction. They used an approach called blending into crowds, i.e., concealing one's query within the queries of other users. To make a web transaction, a user forwards his/her query to a randomly selected peer, the peer then flips a biased coin to decide either to forward a query to another peer user or to the WSE. A path is established between the query initiator and the WSE, the reply comes through the same path and the result is delivered to the initiator as shown in Figure 2.4. For retrieving data privately from a WSE, the Crowds was implemented through a

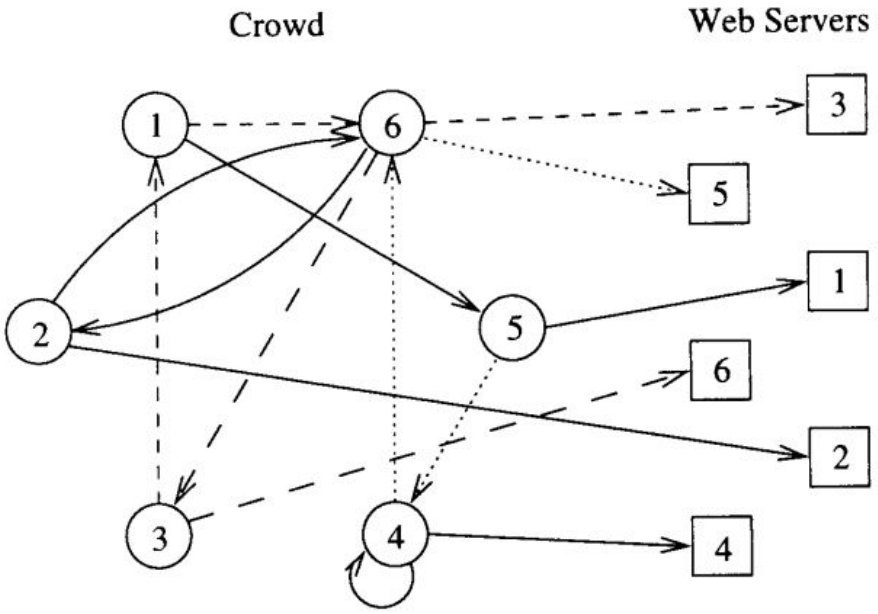

FIGURE 2.4: A path between a user and Web Server in a Crowds [38].

client-side software called jondo and a server-side software called blender. With the aim of achieving confidentiality, the query had to be encrypted using a symmetric key encryption scheme and it had to be exchanged between jondos' before reaching the WSE. Furthermore, Reiter and Rubin presented the degree of anonymity as is shown in Figure 2.5 for analyzing the security achieved by the Crowds. In the year 1998, 1400 copies of crowds (Jondos) were distributed free of cost with active blender for maintaining crowds. The description of crowds project were available online [7]

Weakness: Crowd gained anonymity against the WSE but it did not offer any anonymity against the local eavesdropping. The query is encrypted with the symmetric key when it is forwarded from one Jondo to another. Thus, every node in the path knows the actual query content. WSE can also find the query originator with the collaboration of multiple Jondos. Figure 2.4 represents a path established between the user and WSE through multiple Jondos.

Ostrovsky and Skeith [58] comprehensively surveyed the different PIR techniques in 2007. They suggested that getting the whole database for a single query is impractical, also, this approach makes a user more susceptible to the database.

Castella-Roca, Viejo and Herrera-Joancomartı [40] in 2009 proposed a new protocol called Useless User Profile (UUP) with the aim that it will allow the users

[7] http://www.research.att.com/projects/crowds.

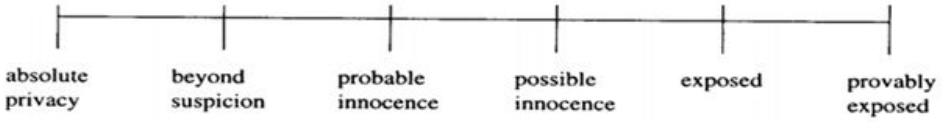

FIGURE 2.5: Degree of Anonymity [38].

to forward queries to the WSE and preserve the privacy by distorting the profile. UUP assumes three entities i.e., users, central server and WSE to preserve the privacy of a user. Group creation, anonymous query sending & retrieval, query forwarding to WSE & result retrieval and result broadcasting are the steps needed in the execution of UUP. The central server creates the group once it has received the requests from predetermined number of users. The connection information is broadcasted in the group so that all users may communicate with each other. Each user is supposed to forward a query to another user so that a query is submitted to the server by another user. Once everyone have exchanged their queries, each user then forwards query to the WSE, When the reply is received from the WSE, the results are broadcasted in the group in clear text. In this way, the WSE is not be able to make a real profile of a user. Moreover, UUP used cryptographic technique (ElGamal group key encryption [59], ElGamal re-masking, and permutation) to preserve the privacy relative to the peer user by hiding the query contents. If each user has forwarded his/her query, the group ceases to exist. In order to send forthcoming queries, the whole process of UUP i.e, group creation, anonymous query sending and retrieval, query forwarding to WSE and result broadcasting repeats again. The group creation puts an extra time delay on the system i.e., 5.2 seconds [11, 34].

Weakness: The UUP has three major weaknesses, (i) the results are broadcasted in clear text, letting all users knew what is being searched inside the group. (ii) UUP fails to provide privacy in the presence of a single adversary. (iii) Once each user has forwarded a query to WSE the group is disassembled and the overhead of the whole process is borne for forthcoming queries.

Lindell and Waisbard [60] investigated and criticized the UUP [40] protocol in 2010. They discussed that UUP [40] is only secure in the presence of semi-honest users that is when all users exactly follow the protocol. But with the presence of even single adversary node, the privacy of all honest nodes can be compromised and

the adversary remains unnoticed. Lindell and Waisbard [60] carried out four types of attacks proposed by [11], (i) Targeted Public key attack. (ii) Stage skipping attack. (iii) Input-Replacement attack. (iv) Input-Tagging attack. With those attacks, Lindel and Waisbard concluded that the privacy provided by UUP [40] can be compromised. Lindell and Waisbard technique work in the following steps: (i) encrypt the query to hide the contents. (ii) the query is shuffled among the users in the group by placing the encrypted query into mixnet according to the technique mentioned in [61]. For the technique to proceed, each user must check if his/her query does appear after shuffling in order to tackle the replacement attack. If the query does appear the user must broadcast the true message in the group. The user can then proceed to decrypt the query and forward it to the WSE. Lindell and Waisbard claimed that even with malicious adversaries their technique achieves security because they do not re-mask the messages like [60] as re-masking is vulnerable to input tagging attack. Instead, [61] used CCA2-secure encryption [62] and onion layered encryption method with the assurance that at least one honest user must "re-mask and permute" all ciphertexts. At the end of the shuffle, each user must verify that his/her encrypted query still appears in the shuffle. When the above-mentioned steps complete, the queries are decrypted and forwarded to the WSE to collect the result. In the end, the results are encrypted using the symmetric encryption algorithm (AES) algorithm and broadcasted to users in the group. Lindell's technique used double encryption to preserve privacy but it puts an extra delay on the system, it is twofold costly as UUP [40].

Weakness: Zhengjun, Liu and Yan [63] has presented attacks that compromise the Lindell's [60] technique and concluded that the privacy of the user can be revealed in the verification stage. When the queries shuffling process concludes, each user has to verify if his/her query still appears after the shuffling. During this time if a malicious user replaces a genuine query of a user and sends a fake query to the WSE, and broadcast the results in the group. The user whose query been replaced upon not receiving an answer to the query will complain and the adversary will link a complaint to find the user. Furthermore, Lindell and Waisbard said that the result of the query will be broadcasted after encrypting through AES algorithm, but they did not explain any method of AES keys exchange.

In 2010 Viejo and Castell'a-Roca [43] used social networks to distort the user profile that was generated by WSE. Their technique has two aims to achieve workable model i.e., to use the social network in which user did not need to generate the groups instead it used the social network to forward queries to the WSE they never used an anonymous channel like TOR because it is a slow approach. Secondly, the WSE should not be able to obtain the profile of the user. Each user had K friends in the social network, the protocol executed in the following steps.

1. The user U_i forwards query q based on privacy function Ψ that estimates the profile exposure level to forward the query to the WSE or its neighbor. However, if the function suggests forwarding the query directly to the WSE, the protocol ends here, otherwise, the query is forwarded to the neighbor.

2. If function Ψ decides that user forward his/her query q to neighbor U_i, U_i either accepts or rejects depending on the level of selfishness. If U_i accepts, he/she follow the step i, if reject U_i selects another friend from the list.

3. When a neighbor U_j accepted a query from U_i he forwards it to WSE in order to get the answer from WSE and forwarded it back to the user U_i. If the query used a path U1-U2-U3-U4, U4 would not be able to link the query to U1. Similarly, U3 and U2 would not know if the U1 was the query q generator or forwarder like them.

Weakness: The social network had a few shortcomings. First, it did not use any encryption and the query was known to all users. Second, if multiple users collaborated, they could find the initiator of the user through the predecessor attack [64].

Erola et al. [44] presented an advanced version of Viejo and Castell'a-Roca's technique [43]. They investigated selfish behavior of the users (the user who request other neighbors to send his/her query to WSE, in turn, the user does not forward other queries to the WSE) in their system using a γ (selfishness function), which decides the level of the selfishness of a user. Every time when a user sends a query to a neighbor, and if he accepts it and answers the query in a specified time, he marks him positive otherwise negative. For sending a query q, user U_i calculates

the sending probability *ps* of all neighbors and executes the user selection function Ψ. The γ function is used to punish selfish users. Their system used the approach of path length 2 before sending a query to the WSE. Additionally, each user will submit the same number of queries as does its neighbor and neighbors of neighbors. The user anonymity is diffused and the true source of the query remains hidden.

Weakness: In this technique [44], If the same static group always submits the same types of queries then WSE can use it to find the real initiator of the query. moreover, this technique is also vulnerable to intersection attack.

Romero-Tris, Viejo and Castell'a-Roca [7] in 2014 extended the work of UUP [40] and introduced a novel idea for anonymous web search in the presence of an untrusted partner. UUP is not secure in the presence of the malicious user, the new protocol improved the security of UUP [40] in the presence of the untrusted user. The new protocol aimed at protecting the privacy of the user against the WSE, dishonest central node and dishonest user. It used the same idea as UUP for group building, ElGamal group key encryption [59] was used for data security. The privacy of a user against the peer user or dishonest user was achieved through Optimized Arbitrary Size (OAS) Benes Network [65]. Benes permutation network is a directed graph, it performs every possible permutation of elements and corresponding inputs and outputs. OAS Benes network actually privately shuffle the queries among the users in such a way that none of the users should be able to link any query with any user. PEP and DISPEP are a zero-knowledge proof technique used for proving that one of the two ciphertexts is the re-masked version of another cipher text. The central node creates a group and gives necessary information to each user for contacting each other such as the IP address and port number. If the central node is dishonest and it groups a single user with n-1 malicious users, then the central node can find the query of the honest user. The WSE is the contact at the last phase of the protocol. Where each user submits someone else's query, the WSE links the query to the user submitting the query. In such a case the profile of the user is obfuscated.

Romero-Tris, Viejo and Castell'a-Roca [66] surveyed all previous techniques of the multi-party peer-to-peer network for private web search. They improved their [40, 44] techniques by reducing the response time to 3.2 seconds, which was 5.8

and 6.8 in previous techniques. The first improvement outperformed all previous dynamic group multiparty protocol with the delay of 3.9 seconds in the simulated environment. In the second improvement, they achieved the privacy of the user between the peer users in the presence of the dishonest user. They used the OAS Benes Network, every time a cipher-text passed a switch, its value is re-permuted and re-masked, so the probability of the user finding the exact user is $\frac{1}{n}$.

In 2018, Domingo-Ferrer introduces the concept of a self-enforcing protocol called co-utile protocol to promote social welfare[45]. The co-utile consists of agents having complementary interests desired to query a WSE but do not want the WSE to learn about his/her interest. The co-utile protocol was modeled as self-enforcing and mutually beneficial for agents (users) using a game-theoretical model for scenario consisting of single-hop query submission game for two agents, single-hop query submission game with multiple agents and multi-hop query submission game. To send a query to the WSE in the single-hop query submission game protocol with two agents (initiator and responder), an agent (initiator) is required to compute the impact of query on his/her privacy using entropy, based on equation 2.1. If the query enhances the privacy of an agent (initiator) relative to the WSE, the agent forwards the query itself, otherwise, the initiator asks the peer agent (responder) to forward the query. The responder first computes the impact of query on his/her privacy based on equation 2.2, if the query enhances the responder's privacy, he/she forwards the query to the WSE otherwise, the responder denies the query-forwarding request. The initiator has to forward the query itself to the WSE.

$$p_I(P) = H(Y_I \cup \{q\}) - H(Y_I) p_R(P) = H(YR \cup \{q\}) - H(Y_R) \tag{2.1}$$

$$p_R(P) = H(YR \cup \{q\}) - H(Y_R) \tag{2.2}$$

Where $p_I(P)$ represents the privacy of initiator agent, Y is the agent's query profile, $H(Y_I)$ is the entropy of initiator's profile, $H(Y_I U\{q\})$ is the entropy after adding query "q" to the agent's profile.

In the single-hop query submission game, the agents would only forward each other queries if their interests were complementary; this assumption limits functionality (having the query answered) and the privacy of an agent. To overcome this limitation, a single-hop query submission game protocol with multiple agents is considered. It works in the same way as a single-hop query submission game protocol with two agents, however, when the responder denies the initiator's request for submitting a query to the WSE, the initiator asks another responder agent to send a query. The initiator added a time threshold to the query i.e. if the query is not answered in time t, the initiator sends it himself, as functionality is the primary focus of co-utile over privacy.

Weakness: The co-utile protocol has two shortcomings, first, there was no privacy between the agents. Every agent was exactly aware of the interest (query) of another agent hence compromising the privacy relative to the responder agents. In such case, a the agent's privacy is solely dependent on responder agent. Second, functionality and delay were the other serious issue associated with the co-utile, a query may not get answered at all or an agent had to wait for long or at the end an agent may have to forward a query on his/her own.

2.5.2 Unlinkability Solutions

As discussed earlier, the aim of the unlinkability solution is to disassociate the query and the user. Some of the prominent unlinkability solutions are discussed below.

In 2008, Domingo-Ferrer et al. [39] introduced the concept of Peer-to-Peer User Private Information Retrieval (UPIR). They considered a community of b users, where each user submits a query on behalf of another user. Their prime target was to preclude the WSE or database from getting the profile of an individual user. Their second aim was to hide the secrecy of a user in front of other peers. The UPIR technique is based on the concept of a shared memory location, where the users record their queries in the shared memory location. A user reads the query from the memory location and forwards it to the database. The user then retrieves the answer from the database, writes it back in the same memory location. The

query-originating user reads the answer, and hence the data is retrieved from the database. To defend confidentiality, the queries and answers are encrypted under the symmetric encryption technique. Users are able to distinguish between the queries or answer when the memory location is decrypted. Based on the above assumption three techniques were proposed.

1. All to all P2P UPIR protocol: To submit a query to the database/WSE a, user must read the memory location m. In the decryption under the symmetric key, five cases may arise.

   (a) If the memory location is free, then encrypt the query and save it in the memory location.

   (b) If the memory location is already occupied with a query, the user sends the query to the WSE, retrieves the answer, encrypts it under the symmetric key and then records it back in the memory location.

   (c) If the memory location contains an answer to the previous query not read by the user who made it waiting for a short random time.

   (d) If the memory location contains a query submitted by the user before but not yet served by anyone waiting for a random short time.

   (e) Contains answer to the previous query submitted to the WSE by some other user. In such a case the user reads the answer of the previous query, encrypts the new query and records it in the memory location.

   Weakness: In the above all to all UPIR protocol, the identity of the user can be hidden from the WSE but all the group members know all queries being searched in the group. Additionally, by the linking of the IP address to the user accessing the shared memory location, the group members can identify the query originator. An external intruder can also find the query contents and the list of users associated with the shared memory location if the single shared symmetric key is leaked.

2. One to one P2P UPIR protocol: In this protocol each user shares a memory location with each of the other users. To submit a query, each user may need to read the memory location m_{ij} Five cases may occur.

(a) The memory location is free. In this case, the user encrypts his/her query and records it in the memory location.

(b) Contains the query submitted by user U_j in this case user U_i decrypts the query, submits it to WSE, gets the answer, encrypts it and record it in the memory location.

(c) Contains answer to the query submitted by U_j that is forwarded by U_i not yet read by U_j then wait for a random short time.

(d) Contains the previous query of U_i who is expecting U_j to forward it to WSE but still pending, in such case the U_i selects another user with another key and go to start again with step i.

(e) contains answer to previous query of U_i in this case U_i reads and answers, encrypts his/her new query and records in the memory location.

Weakness: With one to one P2P UPIR protocol, the user U_j knows exactly that U_i has forwarded the query because both share the common memory location. The trust of privacy is completely shifted to U_j. This approach requires a high number of shared keys, such as for b number of users it requires $\frac{b(b-1)}{2}$ number of keys. Furthermore, this approach suffers significant delay because U_i has to wait for U_j to read the query from the shared memory location and forward it to database/WSE.

3. Configuration based P2P UPIR protocol: A dealer is assumed to create a key pool for the community of b users in a way that v keys are created and distributed into b blocks of size k according to the (v, b, r, k) configuration [67]. The dealer sends a unique block to each user in a secret way through the asymmetric cipher technique. The dealer then deletes the v keys from the memory. For submitting a query to the WSE, the user needs to randomly select one of supposed keys x_{i_j} from the block which the user shares with one of the other users according to the configuration structure. The user U_i follows the same procedure by reading a memory location m_{ij} and decrypt it under x_{i_j}. five situations may arise.

(a) The memory location is empty, then the user encrypts his/her query using x_{i_j} and records in that location.

(b) The memory location contains a query q_j sent by another user and expects someone to forward it to the WSE. In this case, the user decrypts the query, forwards it to WSE, retrieves the result, encrypts the result, and records in that location.

(c) The outcome is an answer to the previously submitted query by one of the peer users and forwarded by another user. However, the answer is not yet read by that user so the user must wait for some random time.

(d) The memory location contains a query submitted by the same user u_i , however, this query is still waiting for some other user to forward it. In this case, the user goes to step 1 and selects another key to forward the query.

(e) The memory location contains an answer to the previous query submitted by the same user U_i. In this case, the user reads the answer of the previous query, encrypts his/her new query, and writes it in the memory location.

Weakness: with configuration based P2P UPIR the users sharing the common memory location know the queries forwarded by that group of users. The identity of the user can be revealed through the intersection attack shown below. The above three protocols contain some weaknesses. First, they are not very secure because they use single keys, meaning anyone knows the queries. While the second requires too many keys and each has to wait for the other users. Viejo and Castell'a-Roca [43] showed some drawbacks in [39], such as that they did not show the memory location requirement for their system. Each user must check or read the memory location in the steady interval of time. According to Viejo and Castell'a-Roca the best response time is 5.84 seconds while they did not include the network time involved in communication, etc.

The previous technique UPIR [39] privacy of user against the WSE, is satisfactory, but the privacy against the peer client cannot be maintained because all the queries have to go through a shared memory location. Klara and Bras-Amoros [67] proposed a technique called an optimal configuration for a peer-to-peer network using combinatory configuration. They used

$(v, b, r, k) - 1$-design. v is the number of users, b is shared memory location, r number of users accessing the memory location and one user accessing k memory locations . They concluded that finite projective-planes are the optimal configuration of the P2P UPIR. The anonymity of the user is diffused among $k(r-1)$ other users in the locality, and $k(r-1)$ is an increasing function of privacy against the database. Concepts like ternary ring and singer cycles are a simple construction of projective planes are also discussed.

Swanson and Stinson [41] discussed the previous techniques of UPIR [67] and highlighted the weakness in the projective plane used for optimal configuration, i.e., when $v = r(k-1)+1$. This would mean that the user has a neighborhood consisting of all other users. In such a case, all neighbors will be forwarding queries to the WSE and only the actual user who is the initiator of the query will not. Thus, the WSE can link the query to him. Another attack introduced by [51] is the intersection attack, i.e., if the attacker (WSE) can analyze the neighborhood of the user submitting the query, and all linked queries, then perhaps the WSE can identify the source of the query. Swanson and Stinson [41] made an intersection attack on stokes and Bras-Amoros [67] $(v, b, r, k) - 1$-design. Suppose $v = 12$ (users), $b = 8$ (shared memory spaces) the following design is acquired for memory space.

S1 = {USR1, USR2, USR3}

S2 = {USR4, USR5, USR6}

S3 = {USR7, USR8, USR9}

S4= {USR10, USR11, USR12}

S5 = {USR1, USR4, USR7}

S6 = {USR2, USR5, USR10}

S7 = {USR3, USR8, USR11}

S8 = {USR6, USR9, USR12}

This is a $(12, 8, 2, 3) - 1$-design. The dual design is: USR1 = S1, S5 USR2 = S1, S6 USR3 = S1, S7 USR4 = S2, S5 USR5 = S2, S6 USR6 = S2, S8 USR7 = S3, S5 USR8 = S3, S7 USR9 = S3, S8 USR10 = S4, S6 USR11 = S4, S7 USR12 = S4, S8. Suppose the above configuration was used, there are a series of three queries

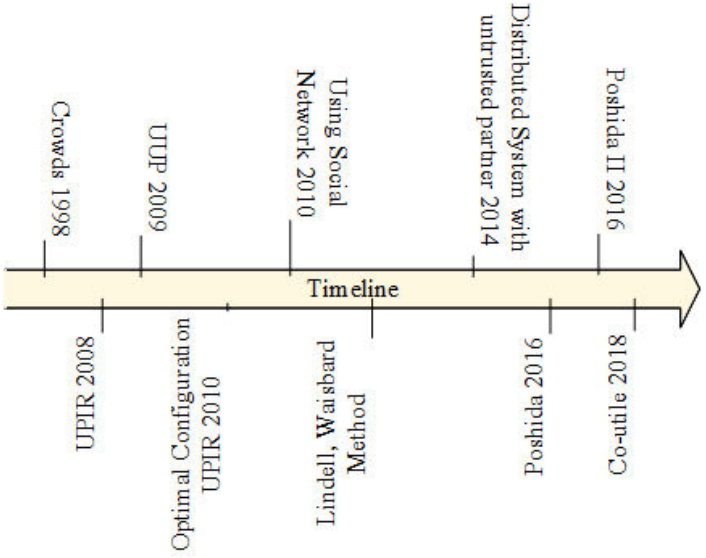

FIGURE 2.6: Timeline of distributed privacy-perserving protocol

on esoteric topics forwarded by USR2, USR8 and USR11 and we want to find the original user in the above configuration. The intersection attack will identify the query originator. If USR2 is the forwarder, then the possible source could be in $S1 \cup S6 = \{$USR1, USR2, USR3, USR5, USR10$\}$. If USR11, then original source might be in $S4 \cup S7 = \{$USR3, USR8, USR10, USR11, USR12$\}$. If USR8, then $S3 \cup S7 = \{$USR3, USR7, USR8, USR9, USR11$\}$. By making the intersection of all possible groups we get USR3 as a source of the query originator. The anonymity of a user among the neighbors can be compromised through algebraic functions. Swanson and Stinson [41] suggested that there are S_l memory locations, each containing k users, and each user is associated with r memory locations. A user must be allowed to forward his/her own queries sometimes with the probability of $\frac{1}{v}$. The user can choose to randomly pick his/her own proxies, while in [67] the user would have to write his/her query into a memory location ,which can be read by another user randomly who forwards it to the WSE. By assigning a proxy to each query, the user can be made anonymous with the attacks (intersection attack) that compromised the privacy.

Kaaniche et al. proposed a decentralized solution CoWSA that empowers end-user to have control over personal data, mitigate single-point failure, ensures the security of the queries, and provides anonymity to a user [68]. User, client, WSE, third parties(TP) and trusted authorities are the five entities of CoWSA. It is basically a proxy solution to retrieve data from WSE on the basis of Sys_Init, Query_Submit,

and Query_Resp procedure. Sys_Init procedure involves interest-based group creation, Query_submit corresponds to the process of sending queries to the WSE by setting a random path through multiple relay user. The Query_Resp occurs when the WSE receives the query aggregates the profile of the user and returns the answer to the user through TPs. However, the CoWSA does not explain how aggregated profiles are computed and what level of obfuscation a client achieve.RSA encryption is used in coWSA to achieve confidentiality.

2.5.3 Implementation of Distributed Protocols

Table 2.1 represents the simulators used in the implementation of various distributed protocols along with simulation platform, dataset used for evaluation, group size, time delay caused by the execution of protocol to retrieve an answer from WSE and encryption techniques. To provide the portability across Unix and Microsoft platforms crowds were implemented in Perl 5. Netscape 3.01 browser was configured to allow 4 simultaneous network connection for the jondos. Apache Web server were used in the implementation of crowds at AT&T lab in the close network proximity.

UUP consisted to two components i.e. Central server (CS) and client application. Those components were implemented through Java programming language. The CS is a multi-thread process that continuously listen to the connection request from the client at TCP port. The client side were implemented through Java applet program to allow the user to search a query in a transparent manner. ElGamal shared key encryption were used to provide confidentiality of query content. The whole UUP system were implemented over a LAN to perform web search secretly. To eliminate the need of CS, the existing social network was employed to achieve web search privacy [43, 44]. They implemented their concept over NS-2 with 400 users [69]. The proposed social network scheme was tested in four social networks consisting of 100, 300, 500 and 1000, where each user was connected to 1-30 users. The proposed social network have not employed an encryption technique when passing query among the neighbours.

The same simulation platform of UUP were employed by UUP(e) i.e. Java programming language were used to implement the protocol. However, the UUP(e)

TABLE 2.1: Simulators used for the implementation of distributed protocol

Protocol	**Simulation platform**	**Dataset**	**Group size**	**Users**	**Delay**	**Encryption Technique**
Crowds [38]	C language and Perl 5 over LAN	static users	4 users	1400	13438ms	SSL
UUP [40]	Java based over LAN	AOL query log	3 users, 4 users, 5 users	1000	5200ms	ELGamal
social network [43]	NS-2	AOL query log	1-10 user	400	3918ms	No encryption
UUP(e) [7]	Java based over LAN	AOL query Log	3 users, 4 users, 5 users	1000	8032ms	ElGamal
Co-utile [45]	LAN	AOL query log	2 users, 3 users, 4 users, 5 users	900	4900ms, 5500ms	No encryption
OSLo [13]	Java based	AOL query log	3 users, 4 users, 5 users	500, 1000	6988ms	RSA

has an extra step of zero knowledge proof. The delay cost of UUP(e) has highest in worst case as compared to the all other distributed protocol.

The co-utile protocol were implemented as multi hope game with 900 agents over LAN. AOL query log were used to compute the privacy through entropy. Co-utile protocol have not employed any encryption or shuffling technique.

OSLo is implemented through Java programming language simulator. OSLo has two component CS and client, the CS is implemented as multi thread process that continuously listen to the connection request from client. Once the CS receive a request from client it records the IP and port numbers. RSA and AES encryption schemes are employed to achieve the confidentiality of query and results. The test are performed with the subset of AOL query log consisting of 500 and 1000 users.

2.6 Summary of Distributed Protocols and Research Gap

The need for query privacy was emphasized in the late ‘90s. Since then many efforts have been made to achieve absolute privacy. Table 2.2 shows a detailed summary of distributed private web search protocols with their strength, weakness

TABLE 2.2: Summary of distributed protocol

Protocol	**Group creation/ Type**	**Query Encryption**	**Limitation and vulnerabilities**		
			Vulnerable to Attacks	**Privacy against peer users**	**Privacy against WSE**
Crowds	Single Group /Static	Symmetric	Predecessor, Denial of Service, collaborating jondos	No privacy peers in path knows the query	Yes But with collaborating jondos can find the user
UPIR	Single group	Symmetric	Input-replacement, Denial of Service	No	Yes
Optimal Configuration UPIR	Single /Multi Groups /Static	symmetric	Input-replacement, Denial of Service, Intersection attack	No	No Local privacy can be compromised
UUP	Single /Multi Groups /Dynamic	No	Denial of Service, Input Replacement, Targeted public-key, Stage skipping, Input-replacement, Input-tagging	No	No Privacy can be compromised through intersection attack
UUP Lindell and Waisbard Improvement	Single /Multi groups /Dynamic	Symmetric	Input-replacement, Denial of Service, Verification stage attack	Yes But can be compromised	No Privacy can be compromised through intersection attack
Social Network	Single Group /Static	No	Predecessor, Denial of Service	No Query is exposed and user can be find with the help of multiple users	Yes
UUP(e)	Single /Multi Groups /Dynamic	Asymmetric	Input-replacement, Denial of Service, Intersection attack, data mining,	Yes (result are broadcast in plaintext)	No can be compromised through intersection attack
Co-utile	Single /static	No	Denial of service	No privacy against peer agents	Yes

and vulnerabilities. Standalone schemes were introduced but the privacy provided by such schemes did not show promising results. Query scrambling is a new technique in query privacy, but so far the result quality obtained are not very good [23].

The peer to peer protocols have charmed the researchers, many efforts have been done on that concept and have settled a pathway. Few protocols like Crowds [38], UPIR [39] gave the initial importance to the problem and guided a way to the solution. UUP [40] introduced distributed concept but lacked security, Lindell and Waisbard [60] added security to [40] but were soon after criticized by Zhengjun, Liu and Yan [63]. Viejo and Castella-Roca[43] proposed that instead of creating groups, social network could be used to achieve the privacy but the predecessor attack [64] without encryption left the content visible. Erola et al. [44] devised a technique to handle the selfish user in social networks through the same static group which again failed to fulfill the promise. Romero-Tris, Castella-Roc and Viejo [7] achieved good privacy for local peers but the WSE was able to link a query

to the user via the association rule mining. However, in the distributed protocols like crowds [38], use social network [43, 44] the query and answer to the query remain visible to the peer users occurring in the path between query originating user and WSE. This compromises the confidentiality and privacy of the user. The protocols like UPIR [39], extended combinatorial design [41], and extended results on privacy [42], uses memory locations to enforce the privacy, but the users associated with memory location can see the query contents and results to the query hence compromising the confidentiality and privacy of the user. In protocols like UPIR [39], UUP [40] and distributed protocol with untrusted partner(UUP extended) [7] the query results are broadcasted in clear text, giving an idea to the users what is being searched inside the group. In the co-utile protocol [45], an agent asks another agent to send a query on his/her behalf to the WSE. In such a case, the peer agents know exactly the queries of the agent. Also, functionality is another prime issue in the co-utile protocol, an agent may ask a peer agent to forward his/her query, but the peer agent may deny the request causing delay to get the answer to the query. In many protocols, users are not allowed to send his/her query to the WSE, giving an attacker to link a query with the originator. Furthermore, the privacy of the user is never evaluated relative to peer users and WSE.

In this dissertation, a series of protocols (OSLo, MG-OSLo, PaOSLo) are proposed to eliminate the limitations mentioned above and to evaluate the privacy of a user relative to the WSE and peer users involved in forwarding queries to the WSE.

2.7 Privacy Evaluation Metrics

Following are the metrics used to evaluate the privacy of users.

2.7.1 Entropy

Entropy is used as a metric for computing the privacy of a user. It estimates the quantity of information represented by a discrete random variable. Entropy provides the quantity of information contained in the probability distribution [70].

In private web search, it measures the amount of information an attacker assigned to a user "X". Let M be the discrete random variable with probability mass function $pr_i = Pr(M = m_i)$, where i represents each possible value that X may take. Entropy is expressed as.

$$H(M) = -\sum pr(m_i).(log_2)pr(m_i) \tag{2.3}$$

where, pr is the probability mass function assigned to the random variable M.

2.7.2 Degree of Anonymity

A degree of anonymity measures the quantity of information the system is leaking [71]. Entropy is used as a tool to calculate the degree of anonymity. Entropy $H(M)$ provides a measure of the average amount of information needed to represent an event drawn from a probability distribution for a random variable. Let M be the discrete random variable denoting the likely query originator in the group. The entropy of M is expressed by Equation 2.3, where $p(x_i)$ symbolizes the probability that the user $p(x_i)$ is the query originator. We denote by H_{Max} the maximum entropy of the system: The degree of anonymity is computed from a ratio of the entropy of a system in the probability distribution (i.e. assigning probabilities to individual user) to the maximum entropy a system can achieve as given in 2.4.

$$d = 1 - \frac{(H_M - H(X))}{H_M} = \frac{H(X)}{H_{Max}} \tag{2.4}$$

Where, H_{Max} the maximum entropy protocol can achieve. $H_{Max} = Log_2(N)$. N is the total number of users in the set.

2.7.3 Profile Exposure Level (PEL)

Profile Exposure Level (PEL) is a privacy metric that measures the percentage of information about a user. It can be measured from the user's obfuscated profile. Authors in [7, 44, 72, 73] have used PEL as a privacy evaluation metric, to evaluate the privacy achieved by the user alongside of WSE. PEL uses mutual information

and entropy to measure the level of user profile exposure. PEL measures the difference between the user's original profile (this profile is built from the set of queries that a user actually generates and send directly to WSE) and obfuscated the profile (the profile obtained when a user executes a privacy-preserving protocol and pollutes his/her profile with the queries of other users). The equation2.5 represents the PEL of a user as a difference of mutual information and entropy.

$$PEL = \left(\frac{I(M,N)}{H(M)} * 100 \right) \tag{2.5}$$

$H(M)$ represents the entropy whereas, $I(M,N)$ denotes the mutual information. Authors in [7, 44] defined M, N as random variables having a sample space ΩM and ΩN. M represent a set of categories of queries that a user actually generated, and N represented a set of categories of queries that the user sends to the WSE. As the user sends many queries other than his/her original query, N commonly contained another user's query categories. PEL measured the percentage of user information promulgation when N is known. The probability function of M, indicated as $pr(m)$ is define as, $\forall m \in \Omega_M, pr(m) = Pr(M = m)$ The Probability function of N indicated as $pr(n)$ is defined as, $\forall n \in \Omega_N, pr(n) = Pr(N = n)$ The conditional probability function of M given N, written as $pr(m \mid n)$. it was defined as $\forall$ m $\in \Omega_M$ and $\forall n \in \Omega_N, pr(m \mid n) = pr(M = m \mid N = n)$ Using probability functions, authors in [44] calculated entropy and mutual information as given in Equation 2.6 and Equation 2.8

$$H(M) = -\sum pr(m_i).(log_2)pr(m_i) \tag{2.6}$$

where, $H(M)$ is the entropy of M, $I(M,N)$ is the mutual information

$$I(M,N) = H(m) - H(M \mid N) \tag{2.7}$$

$$I(M,N) = \sum_{m,n} pr(m \mid n).pr(n)log_2 \left(\frac{pr(m \mid n}{pr(n)} \right) \tag{2.8}$$

$H(M|N)$ is the conditional entropy. $pr(m)$ and $pr(n)$ are the probabilities of each element of M and N proportional to its cardinal. The following notations are used to solve the above equations.

$\Omega_M = \{m_i\}_{i=1}^{v}$ is the set consist of query categories of M.

$\Omega_N = \{n_i\}_{i=1}^{w}$ is the set consist of query categories of N.

$Card_M = \{card_m i\}_{i=1}^{v}$ is the set of the cardinal of each item of M.

$Card_N = \{card_n i\}_{i=1}^{w}$ is the set of the cardinal of each item of N.

$V = \sum_{i=1}^{v} Card_{m_i}$, number of items of set M counting repetition.

$W = \sum_{i=1}^{w} Card_{n_i}$, number of items of set N counting repetition.

Probabilities of each element of M and N are proportional to its cardinal, $pr(m)$ and $pr(n)$ are calculated as:

$$pr(m) = \frac{card_{m_i}}{V},\ where, 1 \leq i \leq v \tag{2.9}$$

$$pr(n) = \frac{card_{n_i}}{W},\ where, 1 \leq j \leq w \tag{2.10}$$

$pr(m \mid n)$ is calculated for two cases where $pr(M = m_i \mid N = n_j)$ is computed for each pair of m_i, n_j where $1 \leq i \leq v$ $1 \leq j \leq w$

1) $n_j \notin M$ then

$$pr(M = m_i \mid N = n_j = \frac{Card_{m_i}}{V}\ where\ 1 \leq i \leq v \tag{2.11}$$

2) When $n_j \in M$ then there is $m_k \in M$ so that $m_k = n_j$

a. if $card_{n_j} \leq card_{m_k}$ then

$$pr(M = m_k \mid N = n_j) = 1 \tag{2.12}$$

$$pr(M = m_k{}' \mid N = n_j) = 0 \tag{2.13}$$

b. if $card_{n_j} > card_{m_k}$

$$pr(M = m_k \mid N = n_j) = \frac{card_{m_k}}{card_{n_j}} + \frac{cadr_{n_j} - card_{m_k}}{card_{n_j}} \tag{2.14}$$

$$pr(M = m_k \mid N = n_j) = \frac{card_{n_j} - card_{m_k}}{card_{n_j}}.\frac{card_{m_k{}'}}{V} \tag{2.15}$$

Suppose a user "X" has four queries ("AOL.com", "myspace", "Europe", and "Writing contest"), when these queries are categorized by ODP, the user original profile i.e, denoted by M at the first degree of ODP hierarchy will contain

a {Computer, Computer, Regional and Arts}. Whereas, when he/she executes a privacy-preserving protocol and sends queries like “Yahoo”, “fruits”, “making candy” and “football”, his/her obfuscated profile denoted by N contains categories like a {Computer, Home, Art, and Sports}. PEL refers to the difference between M and N based on 2.5. PEL measures the percentage of ’X’ original information (M) disclosure when obfuscated profile (N) is known. The value of PEL is between 0 and 100, where 100 means full profile exposure and 0 means no profile exposure. Considering the original and obfuscated query of ‘X’, the entropy of the original profile is calculated using 2.8 is given as H(M)=1.50. Similarly, the mutual information between M and N is computed as I(M, N)=0.75 using Equation 2.7. PEL value will be 50% using the equation. presented in 2.2. Further details related to PEL computation are available in [44].

Chapter 3

ObScure Logging (OSLo).

3.1 Introduction

As discussed in chapter 2, the queries sent by the user to the WSE often contain sensitive information, which makes the query linkable to the user. The search queries contain terms like unique User ID, name, user's employers' details, location, religion, health information, gender orientation, political affiliations, faith, beliefs, social security number etc., which can be termed as user sensitive. The release of such information poses a serious threat to user privacy. To preserve the user's privacy, the unlinkability and indistinguishability are the primary focus of a user in private web search. There are several techniques presented to preserve the web search privacy of a user in the recent past. Techniques like standalone schemes only focus on indistinguishability [48, 50] whereas, proxy servers provide unlinkability. However, distributed schemes achieve both unlinkability and indistinguishability. The user's privacy in the Web search is considered preserved if the following three objectives are attained.

1. The query of the user and its results must remain concealed from the group users.
2. The unlinkability between the user and his or her query must be assured.
3. WSE should not be able to build an accurate profile of the user.

Distributed schemes work by the collaboration of multiple users, where each user forwards a query of another user. To preserve the privacy in distributed protocols, unlinkability (hiding the identity of the user) and indistinguishability (the profile is polluted with queries of group users not with the machine generate queries) are the key objectives for a user to achieve. The group users shuffle a query to hide the identity of a user (unlinkability). In the existing distributed protocols, the query is shuffled using coin tossing [38] or Benes network [74]. Each user in the group forwards a query of another user; hence, the queries of the group users' obfuscates the profile of a user. The standalone schemes like [48, 50] use machine generate queries to obfuscate the profile of a user. However, distributed protocols have the advantage of obfuscating the profile with the real queries. This minimizes the risk of a query being filtered out as machine-generated queries. Minimizing the traffic overhead is another advantage of distributed protocol, as a user only forwards a query of another user instead of forwarding a high number of machine-generated bogus queries.
However, there are several limitations identified in the existing distributed privacy-preserving protocols.

1. Distributed protocols like Crowds, Using Social Network, and Exploiting Social Network [38, 43, 44] establishes a path over multiple users between the query-originating user and the WSE/database. This makes it possible for the intermediate users who relay the query to the WSE/database to see the query contents. Similarly, the result from the query has to come through the same path and hence compromises the confidentiality and privacy of the query and the result contents.

2. Distributed protocols like UPIR, Extended combinatorial construction and Designing privacy enhancing technologies [39, 41, 46] use the concept of the memory location where a block of users is associated with each memory location. To send a query, a user first encrypts a query with the encryption key linked to the memory location and then writes a query into the memory location. Another user from the block reads the query from the memory location and forwards it to the database/WSE. The results retrieved from the database/WSE are written back into the memory location. All users

associated with the memory location can see the query content and result of the query, which compromises the confidentiality and privacy of the user.

3. Results retrieved from WSE are broadcasted in clear text int the extended UUP(e), this makes a group users aware of what is being searched inside the group [7].

4. In co-utile protocol [45], a responder agent/user knows the exact query of the initiator agent hence, compromising the privacy of the initiator agent. Furthermore, functionality (to retrieve an answer to the query) is a prime issue in the co-utile protocol. The responder may deny the query forwarding request of the initiator, it only forwards a query to the WSE if the query is beneficial to his/her privacy, otherwise, the query forwarding request is discarded which causes significant delay.

5. Users cannot submit their query since this gives a clue to the adversary to link the query to the originator by narrowing down the link [7, 39, 40].

Additionally, the existing distributed schemes evaluate the privacy of a user either relative to the peer users involved in forwarding the query to the WSE using probabilistic models or relative to the WSE using privacy metric like PEL, entropy or KL divergence. To the best of our knowledge, there is no framework that evaluates the privacy of a user, relative peer user, and the WSE for distributed privacy-preserving protocols.
To tackle the above-mentioned limitations, this chapter proposes a novel distributed privacy-preserving framework OSLo that answers the research question 1 and research question 2 of this dissertation. The main contribution of this work includes:

1. A distributed privacy-preserving Framework called ObScure Loging (OSLo) is proposed that eliminates the limitations mentioned above in schemes [7, 39–41, 45, 46] and preserves the privacy of the user in a private web search.

2. The proposed framework evaluates the local privacy (unlinkability) and profile privacy (indistinguishability) of a user executing a distributed protocol that comprises of the following steps:

(a) The formal analysis of local privacy of the OSLo, by computing the probabilistic advantages a curious user and a coalition of users have in linking a query with the originator.

(b) The local privacy is also measured through the degree of anonymity.

(c) The profile privacy is evaluated by computing the magnitude of profile obfuscation through OSLo using privacy the metric Profile Exposure Level (PEL).

(d) The impact of group size on local privacy and profile privacy is calculated.

This chapter aims to accomplish three objectives. First, to compute the web search privacy of a user by executing OSLo. second, to compare the privacy and performance achieved by OSLo with the benchmark distributed privacy-preserving protocols extended UUP(e) [7] and co-utile [45], and finally to compute the delay caused by the execution of OSLo.

3.2 ObScure Logging (OSLo)

The proposed framework (OSLo) is a distributed protocol, which makes a group of the 'n' user. Each user forwards queries of the other users of the group. The OSLo consist of the following entities.

1. *User:* An individual who is intendeds to search a query over the WSE covertly.

2. *Core server (CS):* A dedicated machine that supervises the working of the protocol.

3. *Search Query Forwarding Client (SQFC):* A user selected by CS to forward queries of group users to WSE for the specified duration.

4. *Web Search Engine (WSE):* A software system, that is used to search information on internet based on queries.

3.3 OSLo Execution Process

The execution process of OSLo starts when the CS starts listening to the connecting requests. Users who want to perform the web search secretly send a connection request to the CS. When the CS receives 'n' number of requests, it creates a group of n (size of group) users. After creating a group the next step is to select the Search Query Forwarding Client ($SQFC$). The $SQFC$ is supposed to forward the queries of all other users in the group to the WSE and broadcast the results of the query in the group. The CS selects each user of the group as $SQFC$ one by one in round robin fashion. Once the user is selected as $SQFC$, the CS requests it to provide the encryption key. The $SQFC$ creates a pair of asymmetric keys (RSA, 1024 bits keys) and shares the public key with the CS. The CS broadcasts the lists of users in the group including the IP address, the port number of each user and the details of the $SQFC$. When the group is formed and necessary details are shared, a user can send a query to the WSE secretly. All stages of this process is depicted in Figure 3.1 To send a query to the WSE, the user first generates a query, then attaches an encryption key for the result encryption making a query message (QMsg). The user then encrypts the QMsg with the public key of $SQFC$ making it an encrypted query, then attaches a q_ID and achieves an encrypted message($eMsg$). The ($eMsg$) is shuffled among the group users to break the link between the user and query. The process of shuffling starts when the user flips a coin to decide to where to forward the $eMsg$. If the coin flip results in head, the $eMsg$ is forwarded to $SQFC$, otherwise to a randomly selected user from the list of available users in the group. After a few numbers of passes, the $eMsg$ reaches the $SQFC$. The $SQFC$ decrypts the $eMsg$, forwards the query to the WSE, and retrieves the results. The $SQFC$ encrypts the query result with the encryption key of the originating user; after encryption, an $eAnsMsg$ is created. The $SQFC$ broadcasts the $eAnsMsg$ in the group. The user who has the decryption key will decrypt the $eAnsMsg$. Once the $SQFC$ has forwarded the queries of all other users in the group, the $SQFC$ sends a relieving message to the CS. The CS selects the next user as $SQFC$, once each user selected as $SQFC$ by CS and they have completed their tasks, the group ends. Figure 3.2 shows the query sending and result process. Following are the steps required in the execution of OSLo.

1. Connection setup
2. *SQFC* selection
3. Query sending process
4. Query shuffling
5. Query forwarding to WSE
6. Query result processing and broadcasting

3.3.1 Connection Setup

A user who wants to connect to the CS, sends a connection request to the CS. When the CS receives 'n' number of requests, it creates a group having 'n' members. Algorithm 1 shows the server side algorithm, line No. 4 depicts when the CS receives a connection request from a user. The CS accepts the request and registers the IP address and port number of a user making the request. When the group size completes, the CS selects an *SQFC* (line No. 9) by calling a function Select SQFC (line No. 14). The function returns the details of *SQFC* (line No. 18), the CS then broadcasts the user list and the details of *SQFC* (line No. 10). Figure 3.1 shows the users' connection process and SQFC selection process.

3.3.2 SQFC Selection

Each user of the group is selected as *SQFC*. The *SQFC* is supposed to act as a proxy for one query for each user of the group. Figure 3.1 depicts the *SQFC* selection process, When the CS selects a user as *SQFC* from the user_list, the CS forwards a get get_SQFC_info message to the user shown in Algorithm 1 line No. 17. When the user receives the message from the CS, the user calls a $get\_SQFC\_info()$ function (Algorithm 2 line No. 5), and it generates a pair of a RSA 1024 bits of public key & private key. The The $get\_SQFC\_info()$ returns the detailed message *detail_SQFC* to the CS which contains the public key and port number shown in Algorithm 2 line No. 4-13. The CS broadcast the

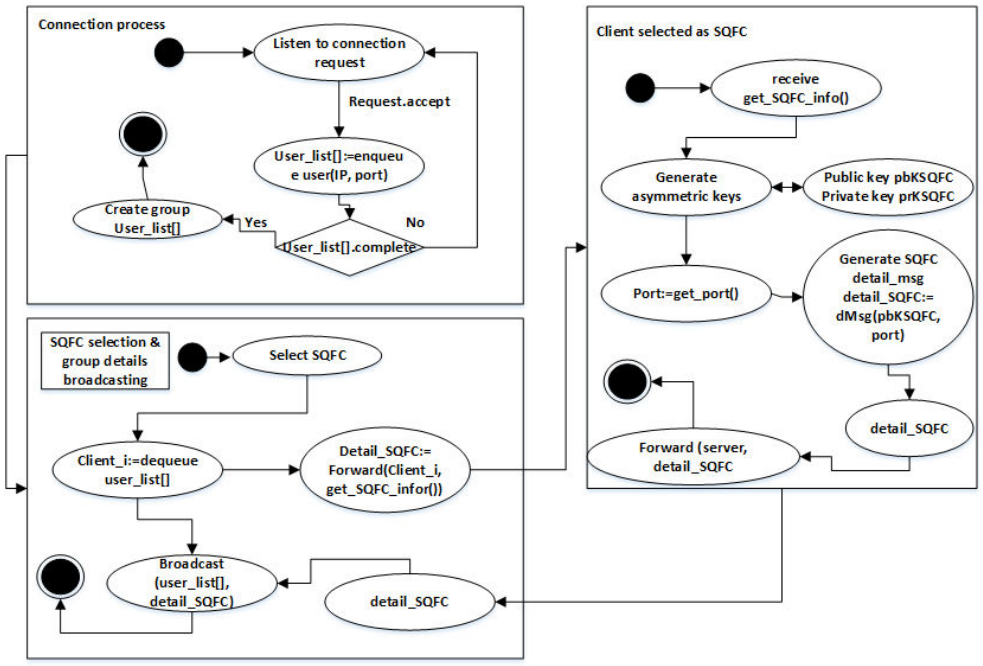

FIGURE 3.1: Activity diagram of user connection and SQFC selection

Algorithm 1 ***OSLo:*** *Server Side Algorithm*

1: **procedure** SERVER SIDE
2: ***Input:***$\{Connection\_request,\ relieved\_msg\}$
3: ***Output:***$\{user\_list,\ detail\_SQFC\}$
4: ***Receive(Connection_request)***
5: $Server := request.accept()$
6: $user\_list[\,] := enqueue\ (get(IP, port))$
7: $if(user\_list.complete)$
8: $counter\_variable = 0$
9: $detail\_SQFC = SelectSQFC(couner\_variable)$
10: ***Broadcast(user_list, detail_SQFC)***
11: $[end\ if\ structure]$
12: ***Receive(relieved_msg)***
13: $detail\_SQFC := SelectSQFC(counter\_variable++)$
14: **Broadcast(detail_SQFC)**
15: ***SelectSQFC(counter_variable])***
16: $Client := dequeue((List\_of\_user[counter\_variable])$
17: $detail\_SQFC := forward(client, get\_SQFC\_info())$
18: $return(detail\_SQFC)$

information of $SQFC$ in the group like IP address, Port number, and public key using line No. 10.

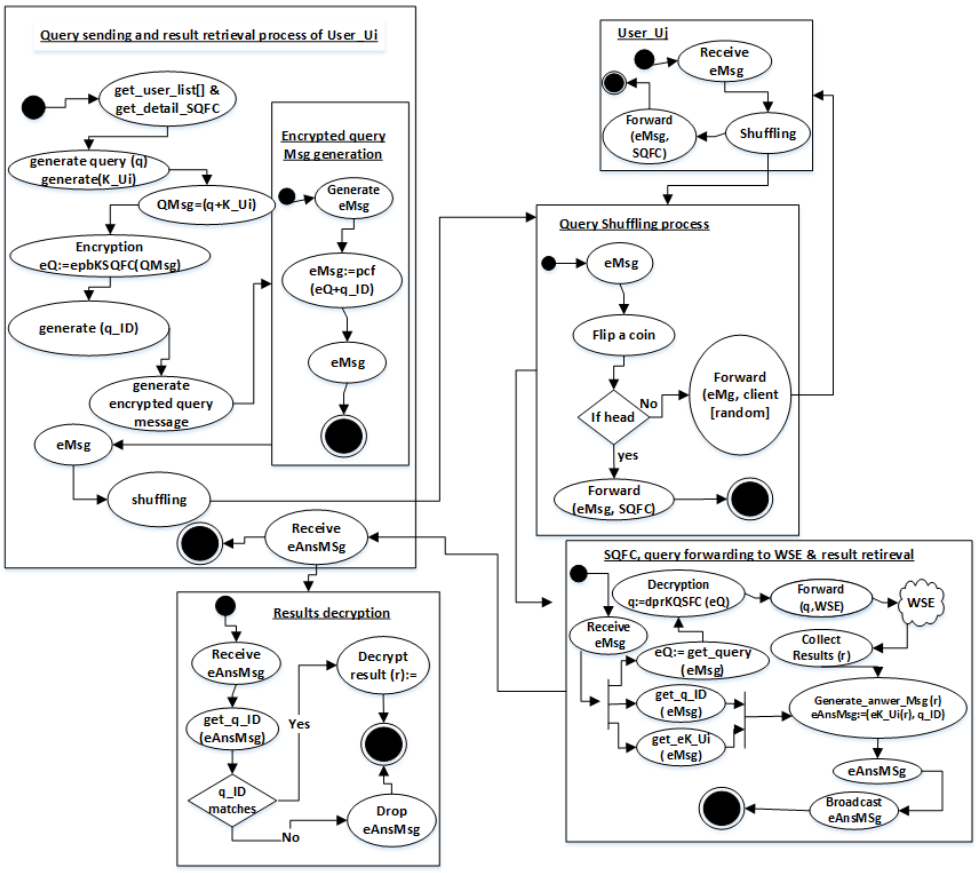

FIGURE 3.2: Activity diagram of query sending and result retrieval process

3.3.3 Query Sending Process

After connecting with the CS, when a user wants to perform a web search secretly, a user is required to get the list of all online users and the information about $SQFC$ (Algorithm2 line No. 15-16). Figure 3.2 shows the activity diagram of the step by step execution of query sending, query shuffling and result retrieval process. In the query sending process the user first gets user_list and the detail (public key and port number) about the $SQFC$. The user generates a query (q), an encryption key (K_U_i) and a random number q_ID. The (K_U_i) will be later used for query result encryption & q_ID will be used for matching. The user then generates a query message (QMsg) by concatenating the query (q) and (K_U_i). Query encryption is the next step of this process, the user encrypts the QMsg with the public key of $SQFC$ making an encrypted query (eQ). In the following step, the user concatenates the eQ) and q_ID using packet creation function (pcf) to generates an enrypted query messge $eMsg$. This whole process is shown in Algorithm 2 line No. 17-28. Once the process of encryption completes,

the $eMsg$ is shuffled between the group users to obscure the identity of a user among the group. The queryshuffling() function is called (line No. 29) and $eMSg$ is passed in the parameter.The queryshuffling() function shuffles the $eMsg$ among the users. The query shuffling function is detailed in line No. 35-45 of Algorithm 2.

3.3.4 Query Shuffling

To obscure the identity of a user the $eMsg$ is shuffled amongst the group users depicted in Figure 3.2. The shuffling function "Queryshuffling(eMsg)" is shown in Algorithm 1 (line No. 35-47). In the shuffling function, the user first gets the details of group users and the information about SQFC (line No. 37-38).

The process of shuffling begins when the user tosses a coin to decide where to forwards the $eMsg$. If the coin produces the head, the $eMsg$ is forwarded to $SQFC$. Otherwise, the $eMsg$ is forwarded to a group user that are selected stochastically from group members. The implementation of coin tossing is shown in Algorithm 2 (line No.40-45). The process begins when a user generates a random number 'X' between 0 and 10 (line No. 36), the user then takes a mod 2 of 'X' in the next line, if the value is equal to 1, we consider this value as a head of the coin tossing, and the $eMsg$ is forwarded to $SQFC$. However, if the 'X' mod 2 equal to 0 (it is considered a tail of the coin tossing) the $eMsg$ is forwarded to a randomly selected group user U_j, from the list of group users. The shuffling ends when the $eMsg$ are forwarded to $SQFC$. The line 46 of an algorithm 2 (Receive(eMsg)) indicates when a user in a group receives an $eMsg$ from another group user during the shuffling process will call the $Queryshuffling(eMsg)$ in the next line (line No. 43) to decide either to forward a query to $SQFC$, or another group user for further shuffling.

3.3.5 Query Sending to WSE and Result Retrieval

When then $eMsg$ reaches the $SQFC$, the $SQFC$ gets the eQ message, the user's encryption key K_U_i and the q_ID. The $SQFC$ decrypts the eQ to get the

Algorithm 2 OSLo: *Client Side Algorithm*

procedure *User Side*
  Input:{get_SQFN_info, $eMsg$, $user\_list$, $detail\_SQFC$, $eAnsMsg$}
  Output:{$eMsg$, $Result$)
  Receive(get_SQFC_info)
    $get\_SQFC\_info()$
      $Counter\_variable := 0$
      $Generate\_Asymmetric\_Keys()$
        $pbKSQFC := get\_public\_key()$
        $prKSQFC := get\_private\_key()$
      $port := get\_port()$
        $generate\_detail\_msg()$
      $detail\_SQFC := dMsg(pbKSQFC, port)$
      $return(detail\_SQFC)$

Query sending process
    $G := get\_user\_list()$
    $eKSQFC := get(detail\_SQFC)$
    $Generate\_Query$
      $q := generate\_query()$
    $Generate\_Key()$
      $K_{\mathrm{Ui}} := get\_eKey()$
    $Generate\_random\_number$
      $generate(q\_ID)$
    $Generate\_Query\_Message$
      $QMsg = concatenate(q, K\_U_{\mathrm{i}})$
    $Encryption()$
      $eQ := eKSQFC(QMsg)$
    $Generate\_eMsg\_packet()$
      $eMsg := pcf(eQ, q\_ID)$
    $queryshuffling(eMsg)$

Result Retrieval process
    Receive(eAnsMsg)
      $q\_ID := get\_q\_ID(dKUi(eAnsMsg)$
      $if(q\_ID.match)$
        $Result := get\_result(eAnsMsg)$

```
35: \\Query shuffling process
36:     Queryshuffling(eMsg)
37:         G := get_user_list()
38:         SQFC := get(detail_SQFC)
39:     Queryshufforwarding(eMsg)
40:         X := generate_random_number(0, 10)
41:             If(x mod 2 = 0)
42:                 y := get_random_user_details(size(G))
43:               forward(y, eMsg)
44:             else
45:               forward(eMsg, SQFC)
46:     Receive(eMsg)
47:         Queryshuffling(eMsg)
```

Algorithm 3 OSLo: *Search Query Forwarding Client Algorithm*

```
1: procedure Search Query Forwarding Client
2:     Input :{eMsg}
3:     Output :{eAnsMsg}
4:         Receive(eMsg)
5:             eQ := get_Query(eMsg)
6:             KU_i := get_user_Key(eMsg)
7:             q_ID := get_q_ID(eMsg)
8:             q := dprKSQFC(eQ)
9:             ForwardMsg_WSE(q)
10:            r := forward(WSE, q)
11:            Generate_answer_Msg(r)
12:                eAnsMsg := generate(eK_ui(r), q_ID)
13:            BroadCast(eAnsMsg)
14:            counter_variable++
15:            if(counter_variable == user_list(size))
16:                forward(server, relieved_msg)
```

query contents. The query contents (q) are forwarded to WSE that processes it and return the results to the $SQFC$, mentioned in Algorithm 3 (line No. 9-10). To achieve the confidentiality of results contents are encrypted with the encryption key (K_U_i)of the user. The $SQFC$ generates an encrypted answer message $eAnsMsg$ by passing the result "r" to Generate_answer_Msg(r) (Algorithm 3, line No. 11-12). The $eAnsMsg$ is then broadcasted in the group (line No. 13). The $SQFC$ increment the counter variable each time the user forwards a query to the WSE. When the $SQFC$ has forwarded "n" queries it sends a relieved_msg to the server shown in line15-16 of algorithm 3. The CS after receiving the $relieved\_msg$ selects

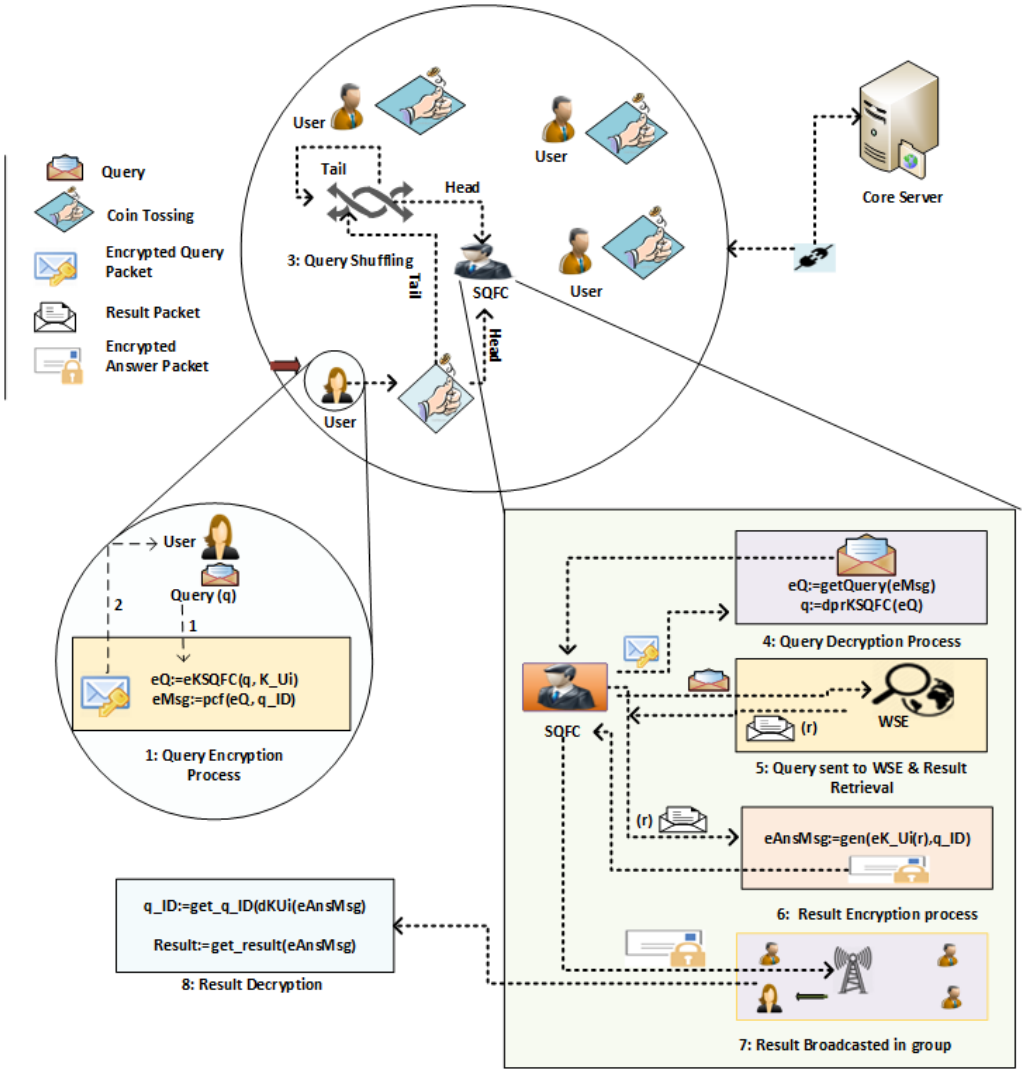

FIGURE 3.3: Graphical representation of query sending process

the next user of the group as $SQFC$ by calling a function SelectSQFC Algorithm 1 (line N.o 15). After getting the details of $SQFC$ the information is broadcasted in the group Algorithm 1 (line No. 14).

3.3.6 Result Decryption Process

When the $SQFC$ broadcasts the $eAnsMsg$ in the group, all users in the group receive the $eAnsMsg$. The user with the decryption key would be able to decipher the result contents. To check if the results is for his or her query the user matches the q_ID. If the q_ID value does not matches the $eAnsMsg$ is dropped by the user. This process of result decryption with symmetric key is given in algorithm 2 line No. 31-34.

TABLE 3.1: AOL query log attributes and description [76]

Attribute	Description
AnonID	Anonymous ID, Distinctively ascertain users in the query log
Query	The query contents, submitted to the AOL search engine.
Query Time	Temporal information including date and time of the query.
ItemRank	Rank assigned to each clicked URL
ClickURL	Address of the clicked URL

3.4 Dataset

This section presents the dataset used in the research. America Online (AOL) released a query log of more than 650 thousand users for the purpose of research [10, 22, 27]. The query log consists of around twenty millions queries, generated by the users in a period of three months from March 2006 through May 2006 [21]. The users were unaware of their queries being released and could be freely accessible[19]. Before releasing the queries, AOL had pseudo-anonymized the query log so that the queries could not be linked back to the originator. The anonymization of query log was achieved by removing all identifiers and personal information such as the name, email address, IP address, etc. The AOL query log dataset consisted of five attributes: i.e., AnonID, Query, Query Time, ItemRank and ClickURL [27]. AOL query log is considered the main experimental data source in the filed of query log privacy [75]. AOL query log has been extensively used in the filed of Web search privacy. Table 3.1 shows the attributes of AOL query log along with its description. Piddinti and Saxena worked on the AOL query log and analyzed different aspect of it, the statistics show that 98.72% have performed less than 100 searches over three months [49]. About 70% of the users have sent less than 30 queries to the AOL log. Figure 3.4 shows the number of users and number of queries relationship. The maximum queries performed by a user in the AOL log is 4960 whereas, around 450K users have sent 10 or fewer queries in a three-months period.

This chapter details the experiments performed to compute the level of privacy (profile privacy) a user achieves by simulating OSLo relative to WSE. The experiments are performed with three-month queries of users selected from the AOL query log. Two datasets consisting of five hundred users and one thousand users

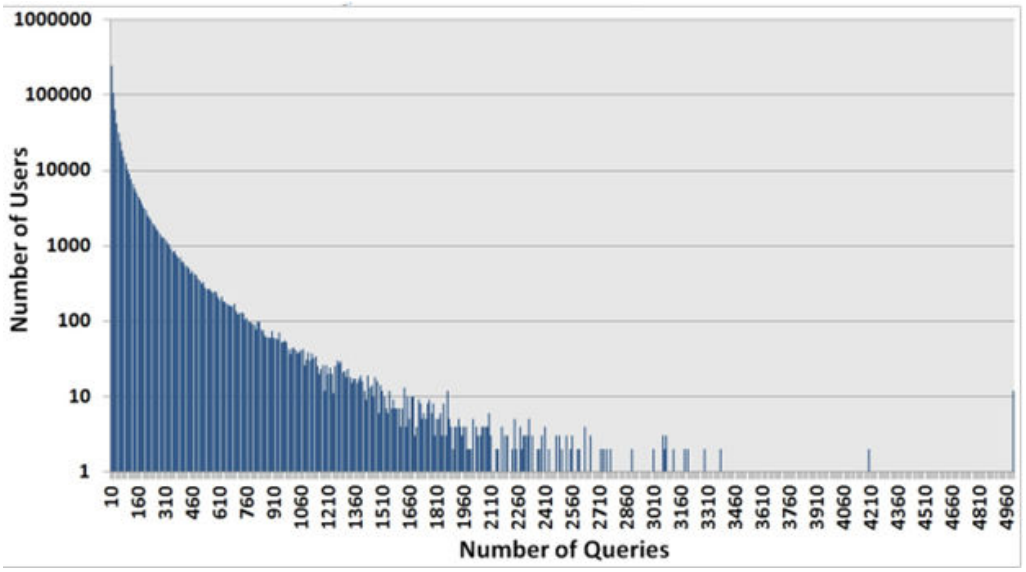

FIGURE 3.4: Statistics of AOL query log by Peddinti and Saxena [49]

selected from the AOL query log are extracted to measure the profile privacy of a user. The description of these datasets is given in the Section below. The reason for selecting two datasets is to find out the effect of the number of users in the dataset on the profile privacy. The experiment over these datasets will validate the impact of varying users in the datasets over the PEL. The difference between the profile maintained by the WSE will dictate the level of profile obfuscation with both datasets. We have used only two attributes of the dataset, i.e., AnnonID and Query. There is some pre-processing applied on the dataset before using it to build the user profile. The pre-processing steps and the profile building process is explained in Section 3.5.3. The dataset used in this work is available at Git-Hub[1].

3.4.1 Dataset 1

As mentioned earlier, in the AOL query log, 98.72% users have sent less than 100 queries whereas, 70% of the users have sent less than 30 queries. The users we have selected in this dataset, have sent between 20 and 661 queries. The users in this dataset are selected randomly; the dataset contains users that have sent at least the average number of queries and those that have sent a good number of queries. The dataset contains 170 of those users who have sent 20 – 50 queries in a three-month period, similarly 55 users have sent up to 75 queries. Furthermore,

[1]https://github.com/mrmohibkhan/dataset

TABLE 3.2: Dataset 1: Range of queries sent by a user

Queries range	Number of User
20-50	170
51-75	55
76-100	50
101-150	60
151-200	38
201-300	50
301-400	30
401-661	47

TABLE 3.3: Dataset 2: Range of queries sent by a user

Queries range	Number of User
20-30	210
31-40	180
41-50	245
51-100	140
101-200	105
201-300	50
300-400	30
400-600	20
600-1515	20

50 users have sent up to 100 queries, and so on. Table 3.2 shows the description of dataset 1 used in the simulation to compute the privacy of the user relative to the WSE.

3.4.2 Dataset 2

The statistical selection of dataset 2 consisting of 1000 users, the description of dataset 2 is shown in Table 3.3. Users are selected randomly from highly active users to least active users. The selected users has sent a minimum of 20 queries up to a maximum of 1514 queries. This dataset contains 210 of those users who have sent 20 - 30 queries, 180 users have sent 31-40 queries, and 245 users have sent 41-50 queries. In this dataset we have 775 of those users who have sent 20-100 queries, 105 users have sent 101-200 queries, 50 users have sent 201-300 queries, and around 100 users have sent 301-1515 queries within a three-month period. The reason behind this is to select users from least active to highly active to see the impact on the profile of those users.

3.5 Privacy Mechanism

As discussed earlier, the primary privacy requirement of a user executing a distributed privacy-preserving protocol is to achieve the following three objectives: i) query contents and result contents remain hidden from the group users. ii) query contents shall not be linked to the originator, and iii) the WSE shall not build the actual profile of a user. To fulfill the above requirements, the OSLo evaluates the privacy of a user in two dimensions, i.e., local privacy to ensure unlinkability and profile privacy to confirm indistinguishability. The local privacy is considered preserved if a user achieves the first and second objectives. Whereas, profile privacy is preserved when the third objective is accomplished. Following are the steps performed in the execution of OSLo:

1. Query Encryption
2. Query Shuffling
3. Query Forwarding to WSE and Result Retrieval
4. Result Broadcasting.

3.5.1 Adversary Model

In any adversary model, the objective of the adversary is to compromise the privacy of a user. In this chapter, any entities involved in forwarding the query to the WSE and retrieving the information on the internet are considered adversary. All entities of OSLo are considered curious, i.e., each entity follows the protocol according to the description mention in Section 3.3. The objective of each adversary entity is detailed below.

1. User: Although user cannot see the contents of a query or results, however, user can collaborate with $SQFC$ to link query with the originator.
2. $SQFC$: The $SQFC$ sees the query content, whereby the objective of $SQFC$ is to link a query to the user. The $SQFC$ can collaborate with other entities to find the query source user.

3. Core Server (CS): The CS supervises the operation of OSLo including group creation. However, the CS does not take part in query shuffling. Additionally, the query is encrypted, however, the CS can make a collation with $SQFC$ or other users to link a query to the user.

4. WSE: The objective of WSE is to build the real profile of a user.

3.5.2 Mechanism to Achieve Local Privacy

The query encryption, query shuffling and result broadcasting are the processes to enact the local privacy of a user. Description of each step is detailed below.

1. **Query Encryption:** Query encryption is used to achieve confidentiality, such that the query contents remains hidden from the peer users involved in forwarding query to the WSE. The user can use a symmetric encryption scheme or asymmetric encryption scheme to achieve confidentiality. The symmetric encryption uses the same key for both encryption and decryption. It can provide confidentiality against the external eavesdropping, however, any user in the group will see the query contents, and hence the confidentiality of the query contents is compromised. Asymmetric encryption has an advantage over symmetric encryption. If the private key would be kept secure, no peer user would be able to see the query contents. Authors in [7, 72, 73] have used RSA and ElGamal shared key encryption schemes.

   In this work, RSA encryption algorithm are used to achieve confidentiality. RSA is an asymmetric encryption scheme and it is easy to share the public key of RSA. Asymmetric keys are best in encrypting data of smaller size, public-key cryptography is usually used for encrypting share keys, digital signatures etc. In this dissertation, the query contents are encrypted with RSA 1024bits public key. The 1024 bit key can encrypt a message of 128 bytes minus any padding or header data (11 bytes) [77]. A typical query is 2.3 words long i.e. around 15 characters, that require around 31 bytes to represent a query [7, 43, 77]. Similarly, an AES key is 128 bits or 16 bytes. The total query message (QMsg) is around 47 bytes including user's

query and user's encryption key(K_U_i). The 1024 bit RSA public key can encrypt QMsg of 117bytes size. In this dissertation, as mentioned in Section 3.7.2 the experiment performed to compute the time delay, the first 30 results returned by the WSE for any query is considered as a potential result. This makes the query result size of around 5kb. Although Arora et. al, [78] performed an experiment to measure the time delay and compare the performance of various encryption algorithms to encrypt text file states that RSA takes an average of 679 milliseconds to encrypt a text file of 2kb, and 748 milliseconds to encrypt text file of 5kb. However, Singh et, al. [79] mentioned that asymmetric encryption techniques are about 1000 times slower than symmetric encryption schemes which makes it impractical when trying to encrypt large amounts of data [79]. Therefore, AES is employed to encrypt the query results retrieved from WSE, as it exhibits a delay of 20ms much faster as compared to the encryption of results through RSA.

2. **Query Shuffling:** There are different techniques available in the distributed protocol to shuffle the query among the group users to hide the identity of query originator. Table 3.4 shows the shuffling methods of existing distributed protocols. The primary aim of the shuffling is to make the query unlinkable with the originator. We are using a coin tossing approach to shuffling query among the group users to achieve local privacy through anonymity. The user generates a random number between 1 and 10, then the user takes a mod of that number if the mod is equal to the one we consider, it as a head, otherwise it is tail. The $SQFC$ forwards queries of the group users to the WSE. The $SQFC$ can see the contents of the query and the result retrieved from WSE. If the $SQFC$ is curious and wants to link a query with the originator, the question is what probabilistic advantages does the $SQFC$ have? The local privacy of the user is evaluated relative to the group entities involved in forwarding query to the WSE i.e., CS, $SQFC$ and group users.

3. **Effects on shuffling using fair and biased coin:** The process of shuffling starts when the user encrypts the query and flips the coin to decide where

TABLE 3.4: Distributed protocol shuffling method

Protocol	**Shuffling method**
Crowds [38]	Coin tossing
UUP [40]	Remasking and permutation
UUP (e) [7]	Optimised Bens Network
UPIR [39, 67]	Encrypted memory location
Co-utile [45]	No shuffling
OSLo [13]	Coin tossing
MG-OSLo	Coin tossing

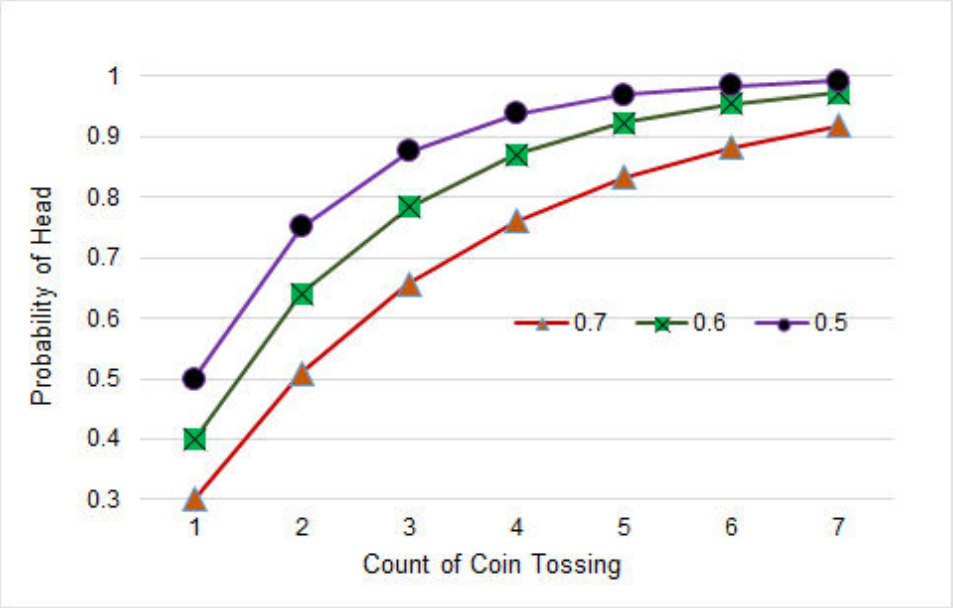

FIGURE 3.5: Probability of Head, After Tossing

to forward the query. If the coin lands on the head, the query is forwarded to $SQFC$, otherwise, it is forwarded to a user randomly selected from the list of all users. Shuffling query obscures the identity of the user among the group peers. The number of time a query is to be shuffled is calculated from the multiplicative rule of probability. Tossing a fair or biased coin affects the number of shuffles. Authors in [38] have used a biased coin to shuffle the query (inside the group). We have calculated the impact of both biased and fair coins on shuffling. Figure 3.5 shows the probability of getting heads after a number of tossing with fair and biased coins. If a fair coin is tossed, the probability that a coin lands on heads on the first toss is 0.5%. The probability of query being forwarded to $SQFC$, on the second, third and fourth attempt is 0.75%, 0.88%, and 0.94%, respectively. However, when a peer wants to shuffle a query more inside the group, a biased coin can be

used. Suppose the probability of forwarding query to a group peer is 0.6%, then the $eMsg$ has to be shuffled five times to obtain a 0.92% chance of obtaining head. It is important to mention that while shuffling obscures the identity of a user among group peers, biased coin can be used only if there are more then three users in the anonymity set (number of users in the group). However, shuffling a query many times has no effect on privacy relative to the WSE.

3.5.3 Mechanism to Achieve Profile Privacy

WSE receives queries from the user and builds the profile accordingly. The profile privacy of the user is considered preserved if the WSE fails to construct the accurate profile of the user. The user executing the OSLo forwards queries of other group users, hence his profile is obfuscated with the real queries of group users. To measure the profile privacy, an experiment is performed that computes the magnitude of profile obfuscation after executing the OSLo. The profile privacy, a user achieves is compared with the benchmark distributed protocol UUP(e) [7] and co-utile [45]. The experiment measures the difference between the user profiles (original profile and obfuscated profile) without executing any privacy-preserving protocol, i.e., sending queries directly to the WSE and after executing the protocols, i.e., forwarding the query to WSE through another user of the group using privacy-preserving protocol (co-utile, UUP(e) and OSLo). Let P symbolizes the original profile of the user built from the queries sent directly to the WSE without executing a privacy-preserving protocol, and Q represents the obfuscated profile build from the queries after executing the privacy-preserving protocol. The primary step of the experiment is to build the profile of a user. The profile building process is explained below.

1. **User Profile Building:** To evaluate the privacy a user achieves relative to the WSE and to measure the level of profile obfuscation, profile building is one of the major steps. The profile of a user is built from the queries a user send to the WSE. This profile is used by the WSEs to retrieve personalize results, also considered as a source of revenue for WSEs. Authors in [7] have

proposed steps to build the user profile. These steps involve the morpho-syntactic analysis and semantic analysis of the queries. Details about each step is described below.

(a) **Morpho-syntactic Analysis:**

The primary step of profile building is to recognize the main topic of the query. In the morpho-syntactic analysis of the query content, Natural Language Process (NLP) techniques based on maximum entropy are used to syntactically analyze the user query. To acquire the main category of the user's query, the NLP techniques like sentence detection, syntactical-parsing, tokenization, stop words removal, stemming and part of speech tagging are followed. The complete description of each step of NLP techniques is described in [80]. By applying these steps, the main topic of the query can be extracted. The next step if the semantic analysis of the query.

(b) **Semantic Analysis:** The keywords acquired in the prior step are sent to DMOZ [2] to discover the hierarchy of query topic. DMOZ is an open-content directory of World Wide Web links, the site and community who maintained DMOZ are also known as the Open Directory Project (ODP). ODP is the largest human editable web directory maintained by a community of volunteers [81, 82]. Figure 3.6 shows the ODP hierarchy categorization, containing around 1 million different categories at ODP, whereby there are sixteen different categories at the top level (first degree). When the user queries are sent to ODP, it categories the user's query into a hierarchy of categories [81, 82]. Any query sent by a user to ODP are categories, e.g., at first degree the query is categorized into one out of 16 categories, at the next degree (second level of hierarchy) the query is categorized into sub categories and so on so forth. Consider a user query "mac.com", the ODP categories this query as "Computers: Software: Operating Systems: MacOS: Internet". The query "mac.com" at first degree is categorized as "computers", at second as "Software", at the third degree as "Operating Systems", at the

[2] dmoz.org (accessed on: 6 February 2017)

fourth degree at "MacOS", and "Internet" at the fifth degree in the ODP directory. Thus, the user whose query is "mac.com" will have computers, software, operating systems, MacOS in his profile. Table 3.5 shows an example of a few queries categorized by ODP into a hierarchy of categories.

The syntactical analysis and semantic analysis are the two analysis applied on the queries of a user. The corresponding profile of the user is built at the first four degrees of the ODP hierarchy. Consider a user 'X' has nine queries such as, snooker, rugby, Java, XML, Honda, Jeep, herpes, Boeing and HIV. When these queries are sent to the ODP for categorization, the ODP categorizes X's queries into a hierarchy of categories as shown in Table 3.6. The profile of 'X' at the first degree contains "Sports, sports, computer, computer, Recreation, Recreation, Health, Recreation and Health". The profile of 'X' at the second degree contains the categories: "Cue, Football, Programming Language, Data formats, Motorcycle, Autos, Condition and disease, aviation, Condition and disease". Similarly, the profile of 'X' at the third-degree contains categories: "Sports, Rugby, Java, Markup, makes and model, Makes and model, aircraft, and immune". In this work, the profile of a user is built for two scenarios, first with the user original queries without executing any privacy-preserving protocol; this profile is called the original profile. In the second scenario the obfuscated profile is built from the queries that a user actually sends to the WSE after executing the privacy-preserving protocol.

3.6 Privacy Evaluation

As discussed, the primary privacy objectives of a user executing the distributed privacy-preserving protocols are presented in Section 3.1. The OSLo evaluates the local privacy to confirm unlinkability and profile privacy to ensure the indistinguishability of a user. The section below describes the results relating to the local privacy and profile privacy of a user. The section below addresses the research question (RQ) 1 of the dissertation.

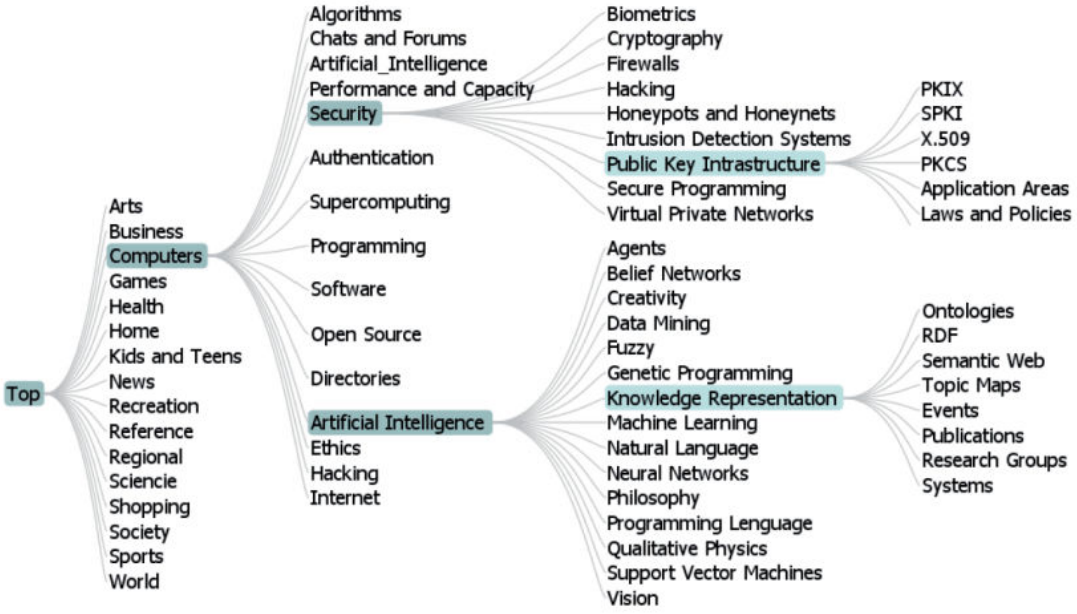

FIGURE 3.6: ODP hierarchy of categories [83]

TABLE 3.5: Example of query categorization by ODP [82]

Query	ODP classification at different degrees
Valley National Bank	Business: Financial Services: Banking Services: Credit Unions: Regional: United States: California
Photography Studios	Arts: Photography: Techniques and Styles: Documentary: Photographers
mac.com	Computers: Software: Operating Systems: MacOS: Internet
Ford fairlane	Recreation: Autos: Makes and Models: Ford: Mustang
PKIX	Computers: Security: Public Key Infrastructure: PKIX: Tools and Services: Third Party Certificate Authorities

TABLE 3.6: Profile of a user X at different degrees

Query	Degree 1	Degree 2	Degree 3	Degree 4	Degree 5
Snooker	Sports	Cue	Sports	Snooker	
Rugby	Sports	Football	Rugby	Union	Clubs and Team
Java	Computers	Programming	Languages	Java	Class
XML	Computers	Data	Formats	Markup	Languages
Honda	Recreation	Motorcycles	Models and make	Honda	
Jeep	Recreation	Autos	Models and make	Jeep	
Herpes	Health	Conditions and Disease	Infectious	Diseases	Viral
Boeing	Recreation	Aviation	Aircraft	Fixed	Wing

RQ1."How to improve the local privacy and profile privacy of a user in a private web search?"

RQ1 (a). How the size of the group and group count affects the privacy and performance of the protocol?
RQ1 (b). What will be the effect of allowing self-query submission and not allowing self-query submission on profile privacy of the user?

Section 3.6.1 gives the result calculations relating to local privacy, and the impact of group size on the local privacy, whereas Section 3.7.1 shows the results relating to the profile privacy answering the second part of research questions.

3.6.1 Local Privacy Evaluation

The local privacy (unlinkability) of the user is considered preserved if no entity (peer users, CS and $SQFC$) able to link query with the originating user. Local privacy of OSLo: As discussed earlier in the query sending process (Section 3.3.3), the user U_i encrypts the query contents with the public key of $SQFC$ and the result retrieved are encrypted with the user's encryption key to attaining the confidentiality. No entity other than $SQFC$ could see the query or result content. However, if the $SQFC$ is curious and wants to link a query to the originator, what is the probability of linking a query with the originator? How much local privacy of a user will be affected if SQFC makes a coalition with the group users? What if the CS is curious, can it link a query to the originating user?
To answer these questions, let two random variables S and P, where S denote the source of the query and P represents the proxy (a peer user in the group) that passes the query to the $SQFC$. Suppose there are 'n' users in the group. If the $SQFC$ has received a leery query on esoteric topic, and the $SQFC$ wants to find the originator of that leery query. The probability of linking the query to the user "U_i" is given below.

$$Pr[S = U_i \mid P = U_j] = \frac{Pr[P = U_j \mid S = U_i] \cdot Pr[S = U_i]}{Pr[P = U_j]} \tag{3.1}$$

$$Pr[P = U_j \mid S = U_i] = Pr[P = U_j] \tag{3.2}$$

$$Pr[S = U_i] = \frac{1}{n-1} \tag{3.3}$$

Where, n represents the number of users in a group and $i, j \in (1...n)$, as $SQFC$ is not the query source so $SQFC$ excludes himself.
Equating (3.2) and (3.3) we get

$$Pr(S = U_i \mid P = U_j) = \frac{1}{n-1} \tag{3.4}$$

Equation (3.4) shows the probability of linking the query to the user by $SQFC$ depends on a number of users in a group, all users in the group are equally probable. However, If $SQFC$ and C users collaborate to identify the query originator then the probability of linking query is given in (3.5)

$$Pr(S = U_i \mid P = U_j) = \frac{1}{n-C} \tag{3.5}$$

Equation 3.5 shows that if $SQFC$ forms a coalition with the C user for the probability of linking the query to the initiator $\frac{1}{n-C}$, which means all compromised C users will be excluded from the list. If C is equal to n, all users are compromised and there is no user to attack, i.e., to whom to link the query. If $n - C$ is equal to 1, it means that all users are compromised except the query generating user, in which case $n - C$ shall be greater than 1.

The CS and peer users cannot read the content of the query and query results as they are encrypted. However, if any of the curious entity makes a coalition with $SQFC$, then equation (3.5) shows the probability of associating the query with the originator. As the CS is not involved in the query shuffling process, and if SQFC does not collaborate with CS or group user, none of the curious entity would see the query or query result and the probability of relating the query to the originator is $\frac{1}{n}$, i.e., all users are equally probable. However, if CS makes a coalition with $SQFC$, the probability of linking the query to the originator are given in (3.4). The group users do not see the query (q) or result returned (r), however, if the compromised peers forms a coalition with $SQFC$ then the probability of linking the query with the originator is given in (3.5)

- Degree of Anonymity

  Anonymity is defined as the state of being unidentifiable within the set of subjects [46]. The maximum anonymity a user can achieve when the attacker sees all other users in the group is equally probable. According to [71] the degree of anonymity depends on the probability distribution, i.e., the probability assigned by the attacker to the individual user. In the proposed protocol, the user forwards a query to $SQFC$ or to a member of the group based on the result of coin flips. If the coin flip produces a head, the query packet is forwarded to $SQFC$, otherwise the query packet is forwarded to another group user selected randomly from the list of all users. When the $SQFC$ receives a query packet from the user, the probability the user forwarding the query packet is that the originator depends on the probability of forwarding. Suppose $SQFC$ is curious and wants to link a query with the originator, $SQFC$ can assign different probabilities to a user based on the type of coin tossing (biased or fair coin). If a fair coin is tossed during the query-shuffling phase of OSLo, i.e., probability of forwarding (pf) query to $SQFC$ 50% and to a randomly selected user is also 50%. When $SQFC$ receives a query from a userU_i there is a 50% chance that U_i is the query originator and a 50% chance that U_i is the forwarder. If there are 3 users in the group apart from $SQFC$, there is a 50% chance that U_i is the originator while there is a 25% chance for each of the remaining two users. In such a case, the degree of anonymity according to equation 2.4 is 89%. Figure 3.7 shows the degree of anonymity and the number of users in the anonymity set relationship. If a fair coin is tossed (pf=50%), the degree of anonymity drops when the number of users in the anonymity set increases. When there are eight users in the anonymity set, the degree of anonymity drops below 80%. However, if a biased coin is tossed (pf=60%), the degree of anonymity is around 86%. Similarly, when the pf=70%, the degree of anonymity for eight users is 93%. Therefore, if there are more than eight users in the anonymity set than it is recommended to toss a biased coin (pf 60%) instead of a fair coin to keep the degree of anonymity above 80% as according to [71] the system should provide the degree of anonymity greater than 80%.

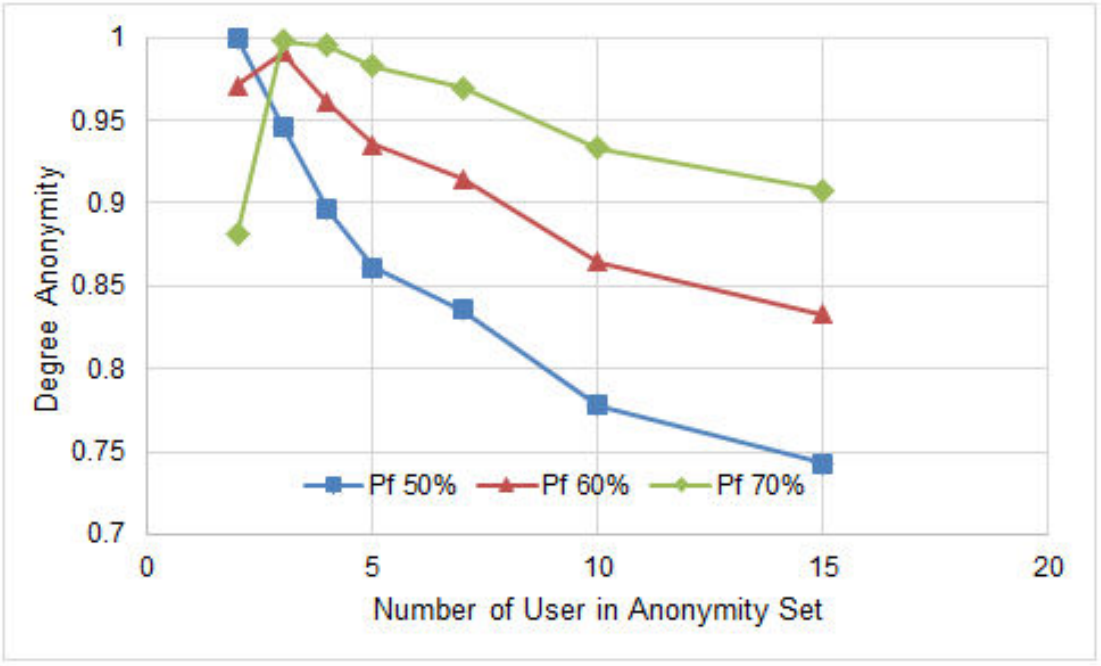

FIGURE 3.7: Probability of Head, after number of tossing

Local privacy of Co-utile protocol:

The co-utile protocol offers no local privacy. A user/agent when requests a peer user to forward his/her query to the WSE sends the query directly to him/her without any anonymous technique or query shuffling. Hence, the peer user who forwards a query to the WSE knows the exact query of the user. In the co-utile protocol, a user shifts his/her trust from WSE to the peer users, if the peer user is dishonest the privacy of a user requesting a query is fully compromised. Furthermore, functionality is another concern in the co-utile protocol. Sometimes, when a user requests a peer agent to forward a query to WSE maybe denied request because the query may not be beneficial for the obfuscation of the peer's profile.

3.7 Results and Discussion

This section gives a detailed discussion of the performance analysis of OSLo in terms of profile privacy and delay.We have compared the OSLo with a co-utile protocol [45] based on privacy and with the latest extended version of UUP(e) [7] (a benchmark distributed privacy-preserving protocol) based on privacy & delay. A Java-based simulator is developed to simulate the OSLo, co-utile and UUP(e)

protocols using a "Dataset 1 & 2" given in Table 3.2 and Table 3.3 and the simulation description given in Table 3.14. The simulation code, actual query log generated after simulating the protocols are available at Git-Hub[3] [4] [5].

3.7.1 Profile Privacy Evaluation

As explained in the introduction, the profile privacy of a user is considered preserved when WSE fails to build the reliable/original profile of a user. To measure the magnitude of profile privacy a privacy metric Profile Exposure Level (PEL) is used to measure the profile obfuscation a user achieves in front of the WSE. The results of profile privacy validate the percentage of disclosure of information for an original profile from the observance of the obfuscated profile. The PEL calculates the difference between the original profile P (built from user queries without simulating the protocol) and obfuscated profile P' (after simulating OSLo). The proposed protocol (OSLo) is simulated with two subset of AOL dataset consisting of 500 users and 1000 users, the details of each subset of users are explained in Table 3.2 and Table 3.3. OSLo is simulated for two situations. First, when self-query-submission is allowed, i.e., when the user forwards his/her query along with the queries of group peers. Second, when query-self-submission is not allowed, i.e., when the user only forwards the queries of peer users but not of his/her query. The results of the former situation is given in Figure 3.8 and Figure 3.9when OSLo is simulated over a Dataset1 and Dataset2 for the group size of 3 users, 4 users, 5 users, 7 users, 10 users and 15 users. Whereas, in the later situation Figure 3.12 and 3.13 shows the average PEL results of a user simulated over the same datasets. Furthermore, the profile privacy results of OSLo are compared with the co-utile protocol and with the latest version UUP(e). The co-utile protocol is simulated for the group size of two users, 3 users, 4 users, and 5 users. Whereas, the UUP(e) is simulated for the group size of 3 users, 4 users, and 5 users. To show the realistic comparison between these protocols, OSLo and co-utile are compared for a situation when self query-submission is allowed because, in the co-utile protocol, a user is allowed to submit a self query if the query obfuscates the person's profile.

[3] https://github.com/mrmohibkhan/OSLo
[4] https://github.com/mrmohibkhan/UUP(e)
[5] https://github.com/mrmohibkhan/Co-utile

Moreover, the latest UUP(e) [7] and OSLo are compared for a situation when self query-submission is not allowed, as a person does not forward a self query in the latest UUP(e). The vertical axis of Figure 3.8 - 3.13 shows the average PEL between 0 and 100. The "0" means the no profile exposed whereas, the "100" means fully profile exposed. The percentage at the vertical axis represents the extent of original information disclosed by the observance of obfuscated information. The horizontal axis represents the ODP hierarchy from degree 1 to degree 4. The ODP categorize the user query into a hierarchy of categories, where the degree 1 represents the more general category many diverse queries fall into the same category at degree 1, e.g., Cricket and Soccer both belong to sports category. However, the higher degrees are more and most specific categories of a query.

1. **Profile Privacy: Self-Query Submission Allowed:**

   Table 3.7, Table 3.8, Figure 3.8, and Figure 3.9 shows the result obtained for the simulation of OSLo when the self-query submission is allowed. The average PEL represents the leakage percentage of the real profile from the observance of obfuscated profile. The result indicates the difference between the uncertainty of the WSE before and after getting the obfuscated profile.

   (a) OSLo PEL with Dataset 1

   Figure 3.8 and Table 3.7 presents the simulation result of the average PEL of a user when selected as $SQFC$ to forward his query and the queries of other users. The average PEL at the first degree of the ODP hierarchy for a group size of 3 users is 67.59%. Similarly, the results depict that the average PEL for a group size of 4 users is 64.59%. When the number of users in a group is increased to 5, 7 and 10 users the result of average PEL is 63.6%, 60.5% respectively. Furthermore, the simulation result depicts the average PEL at degree 2 of the ODP hierarchy for the group size of 3 users to 7 users are 42.6%, 36.33%, 32.5%, 27.94%, 24.31%, and 21.24% respectively. The results show that the average PEL drops when the number of users in the group increases from 3 to 10. This is because the profile of the user is obfuscated with the queries of a higher number of users. However, when the number of users increased from 10 to 15, an insignificant change in the average

TABLE 3.7: OSLo average PEL self-query submission allowed, dataset 1

Number of users	Degree 1	Degree 2	Degree 3	Degree 4
3 Users	67.60	42.60	38.30	37.85
4 users	64.59	36.33	31.03	30.99
5 users	63.61	32.50	26.72	26.30
7 Users	60.74	27.94	21.28	21.22
10 users	58.86	24.31	17.74	17.48
15 User	58.19	21.24	14.54	14.34

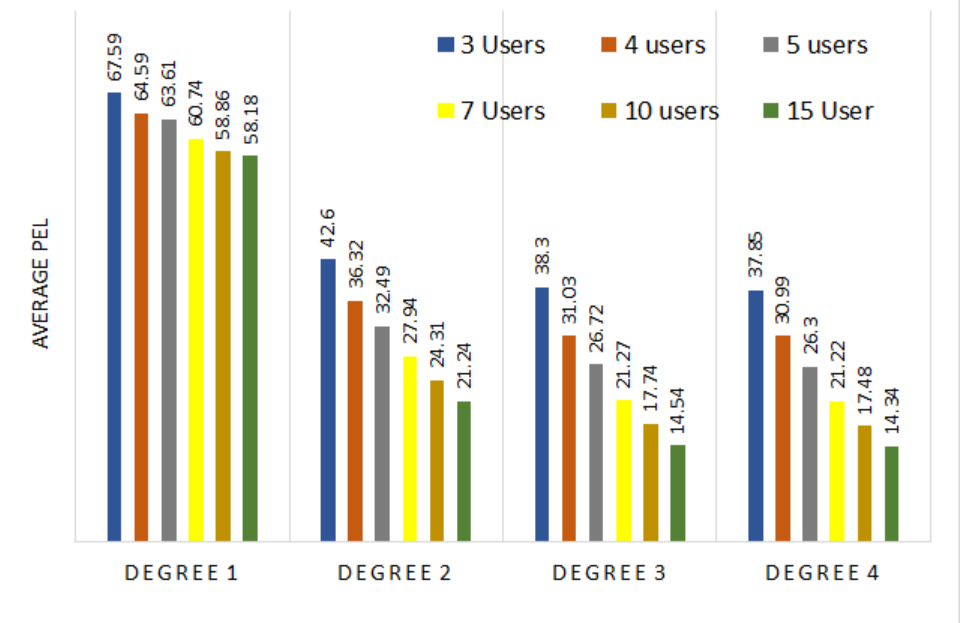

FIGURE 3.8: Average PEL of OSLo with Dataset 1

PEL is observed. As the profile is obfuscated to the maximum value and increasing more users in the group have no significant impact. It is important to mention that depending on the group size, the number of self-queries forwarded by a user is as follows: When the OSLo is simulated with a group size of 3 users, a user when selected as $SQFC$ forwards 33% of self-queries in addition to the queries of other group users. Consider a user has 40 queries to forward to the WSE, by executing the OSLo when self-query-submission is allowed for the group size of 3 users, the user forwards 13 queries of his own and 27 queries of the other group users. Similarly, a user forwards 25% of self-queries along with the queries of group peers with a group size of 4, 20% with a group size of 5, 14.5% with a group size of 7, and 10% with a group size of 10. When the self-query submission is allowed the attacker cannot exclude the user from the group being an originator of the query.

TABLE 3.8: OSLo average PEL self-query submission allowed dataset 2

Number of users	Degree 1	Degree 2	Degree 3	Degree 4
3 users	62.69	40.35	36.91	36.82
4 users	59.72	33.36	29.42	29.51
5 Users	57.89	29.28	25.09	25.27
7 Users	55.35	24.51	19.82	19.89
10 Users	54.48	21.27	16.31	16.44
15 Users	53.49	18.76	13.36	13.21

(b) OSLo PEL with Dataset 2

Figure 3.9 and Table 3.8 shows the result of average PEL of OSLo simulation with Dataset 2 for the same situation i.e., self-query submission is allowed. The PEL at the first degree is 62.69 % for the group size of 3 users, the average PEL decreases when the number of users increases in the group. The value of average PEL drops to 59.71% for the group size of 4 users, 55.34%, 54.48% and 53.49% for the group size of 5, 7 and 10 users. The ODP first degree of a query represents a general category, many dissimilar queries may belong to the same category, and, e.g., queries like eczema, arthritis and pregnancy belong to the health" category at first degree in the ODP ontology. However, ODP higher degree represents the more specific category of a query. The PEL at the second degree of ODP is 40.34%, 33.36%, 29.27%, 24.51% and 21.27% for the group size of 3, 4, 5, 7 and 10 users. Similarly, the degree three and degree four shows the group size inversely affects the PEL. The OSLo when simulated with dataset 2, obfuscates the user profile more as compared to the Dataset 1. The reason of more obfuscation with dataset 2 is the chances of grouping with distinct user get high, the dataset 2 have a higher number of users (1000) as compared to the Dataset 1 which have less number of users (i.e., 500).When the number of users increases the chances that a user may grouped with other users having variety of interests that increases higher obfuscation for dataset 2 as compared to dataset 1.

The profile privacy of OSLo for a situation where self-query submission is allowed cannot be compared with the UUP(e), as the user of UUP(e) never forwards his/her query to the WSE. Additionally, once a user has

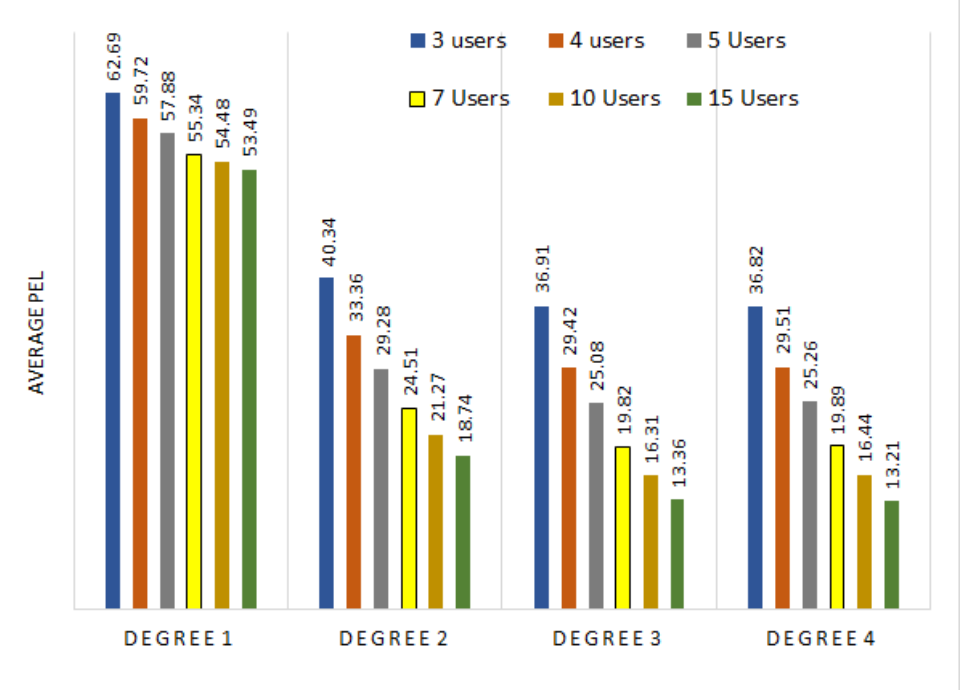

FIGURE 3.9: Avearage PEL of OSLo with Dataset 2

forwarded one query to WSE the group ends, to send another query the whole process of query sending repeats, however, in case of OSLo a *SQFC* forwards "*n*" (where n is the size of a group) queries to the WSE.

(c) Co-utile Average PEL

The working of the co-utile protocol is detailed in section 2.5.1. Table 3.9 and Table 3.10 shows the simulation results obtained for the co-utile protocol over dataset 1 and dataset 2 respectively. The average PEL at degree 1 of the ODP hierarchy for two users is 82.81% for dataset1 and 81.87% at dataset 2. Whereas at the second degree the average PEL is 76.05% & 71.30% at the degree 2 of ODP hierarchy. Similarly, for single hop query submission with 3 users (multi-agent) the average PEL at degree 1 of ODP hierarchy is 74.58% & 75.11%, whereas, at the higher degree (i.e., degree 2, degree 3 and degree 4) the average PEL is 64.21% 62.47% and 63.41% for dataset 1 and 64.74%, 63.41% and 63.85% for dataset 2. Unlike OSLo, the simulation results depict that the increasing the number of users in the dataset has no positive impact on average PEL of a user executing co-utile protocol, instead, the profile of a user is exposed more with it. The co-utile protocol is

TABLE 3.9: Co-utile protocol: Average PEL, self-query submission allowed for dataset 1

Dataset 1				
Number of users	Degree 1	Degree 2	Degree 3	Degree 4
Two users	82.81	76.05	74.92	75.09
Three users	74.58	64.21	62.47	63.41
Four users	74.09	64.02	62.48	63.32
Five users	73.54	61.38	59.47	60.42

TABLE 3.10: Co-utile protocol: Average PEL, self-query submission allowed for dataset 2

Dataset 2				
Number of users	Degree 1	Degree 2	Degree 3	Degree 4
Two users	81.87	71.3	70.09	70.62
Three users	75.11	64.72	63.41	63.85
Four users	77.16	63.45	61.42	61.55
Five users	78.82	64.7	62.74	63.19

the self-enforcing protocol, a user only forwards the query of a peer user if it obfuscates his/her profile otherwise the query forwarding request is denied resulting user has to forward a query on their own, making more exposed to the WSE.

(d) Profile privacy comparison of OSLo VS Co-utile

Figure 3.10 and Figure 3.11 show the profile privacy a user achieve by executing OSLo and co-utile protocol. The simulation results indicate that OSLo preserves 9.37% better privacy as compared to co-utile for the group size of 3 users at degree 1 of the ODP hierarchy when simulated over dataset 1. The percentage raises to 28.97%, 32.40%, and 34.27% at degree 2, degree 3 and degree 4 of ODP hierarchy. Similarly, for the group size of 4 users, OSLo preserves 12.81%, 37.37%, 42.45%, and 43.64% privacy at degree 1-4 of ODP hierarchy as compared to the co-utile protocol. Results for the group of 5 users depicts that the OSLo preserved better privacy as compared to the co-utile. Similarly, when both protocols are simulated with dataset 2, OSLo preserved 16.55% better privacy as compared to the co-utile protocol at degree 1 for the group size of 3 users. Likewise, OSLo preserved 32.49%, 35.33%, and 36.04% better privacy at degree 2-degree 4 of ODP hierarchy as compared to the co-utile protocol. Furthermore, OSLo maintained better

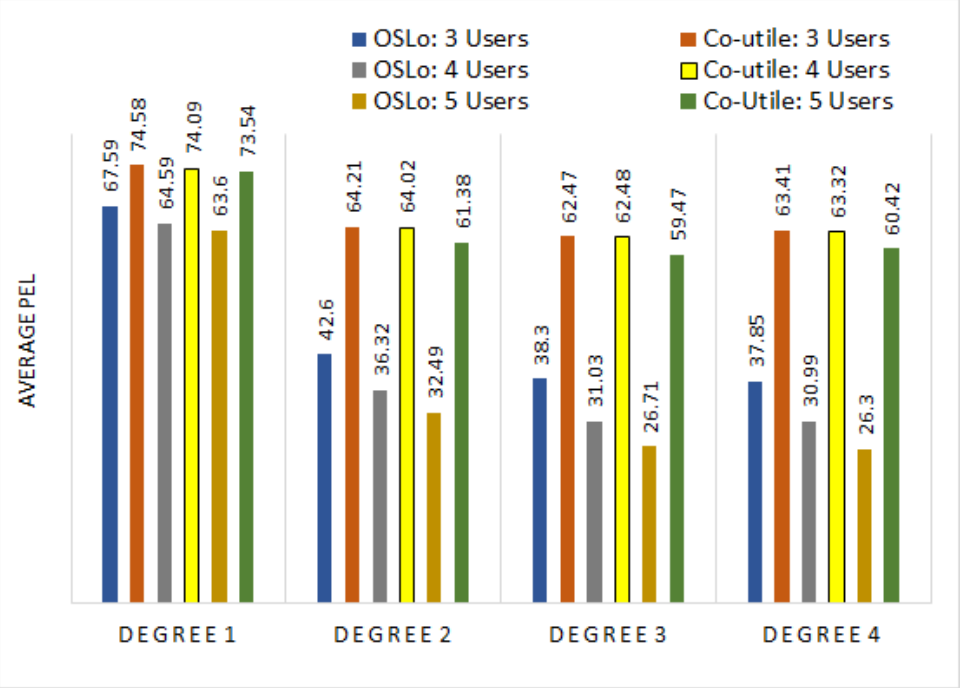

FIGURE 3.10: Avearage PEL of OSLo VS Co-utile with Dataset 1

privacy as compared to the co-utile protocol for any group size. In the co-utile protocol, the user (forwarding agent/user) only forwards the query of initiator if it obfuscates the profile of the forwarding user otherwise the query is denied. In such a case when the forwarding agent denies the request, the initiator has to forward the query on his/her own, resulting in the profile of the initiator is not obfuscated. However, in the case of OSLo, the SQFC has to forward the query of all other users, hence, obfuscating the profile of the user. Functionality (to retrieve an answer to the query) is another prime issue in the co-utile protocol, the responder may deny the initiator's request causing a notable delay in the query answering.

2. **Profile Privacy: Self-Query Submission not Allowed:**

Figure 3.12, Figure 3.13, Table 3.11 and Table 3.12 shows the average PEL of a user executing the OSLo for a situation when the self-query submission is not allowed for the group size 3 users, 4 users and 5 users. The profile privacy provided by OSLo for the above-mentioned group sizes is compared with the state-of-art privacy-preserving protocol UUP(e) as the self-query submission is not allowed.

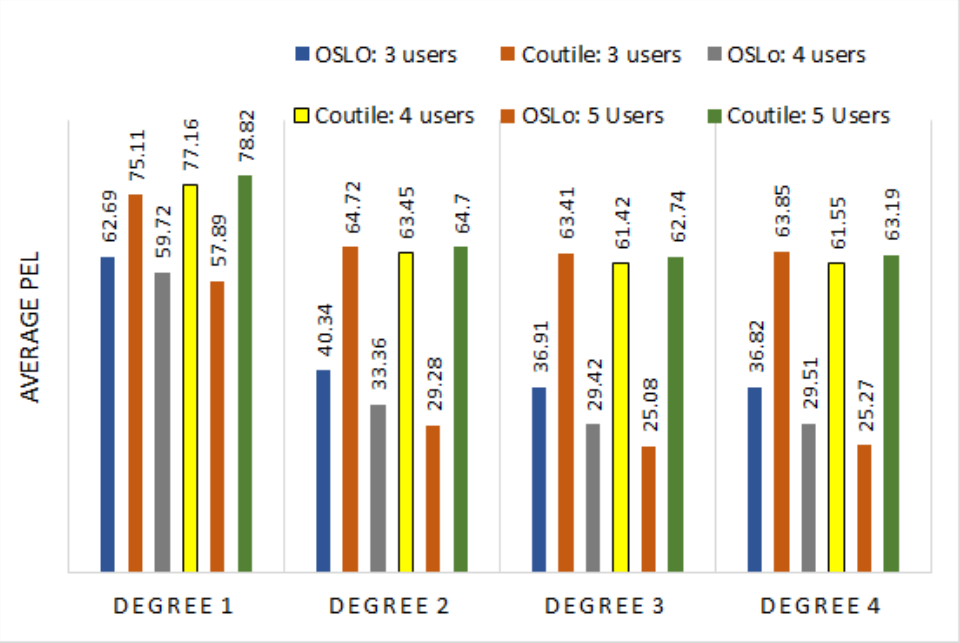

FIGURE 3.11: Avearage PEL of OSLo VS Co-utile with Dataset 2

TABLE 3.11: Average PEL of OSLo VS. UUP(e) self query-submission not allowed dataset 1

Number of Users	Protocol	Degree 1	Degree 2	Degree 3	Degree 4
3 users	UUP(e)	61.37	18.3	8.97	8.82
	OSLo	57.3	16.54	8.86	8.49
4 users	UUP(e)	60.88	17.21	9.08	8.71
	OSLo	56.75	15.68	7.6	7.46
5 users	UUP(e)	57.16	17.17	8.99	8.73
	OSLo	54.18	14.64	6.75	6.38

(a) OSLo average PEL with Dataset 1

Figure 3.12 shows the average PEL simulation result of a user executing OSLo for group size of 3 users is 57.3%. Similarly, the average PEL for group size of 4 users and 5 users are 56.75% and 54.18% at degree 1 of the ODP hierarchy. Likewise, for the second degree of the ODP hierarchy, the average PEL of the user for group size of 3, 4 and 5 users are 16.54%, 15.68% and 14.64% respectively. The PEL at higher degree, i.e., at degree 3 and degree 4 are less 10% for all group size.

(b) OSLo average PEL with Dataset 2

Figure 3.13 depicts the user average PEL result of OSLo simulated with Dataset 2. When there are 3 users grouped together, the average PEL of OSLo for degree 1 of the ODP hierarchy is 47.69%. Likewise, for a

group size of 4 users the average PEL is 48.56%; similarly, the results for the group size of 5 users is 49.15%. The average PEL at degree 2 of the ODP hierarchy for the group size of 3 users, 4 user, and 5 users is 12.78%, 12.76% and 12.61%. The average PEL at higher degrees is less than 8% for any group size.

- Profile Privacy of UUP(e): UUP(e) was proposed to preserve the privacy of a user relative to the WSE. UUP(e) consists of entities like Central server (CS), users and WSE. When the UUP(e) employee single dynamic group sends a query to t he WSE, the user is required to connect to the CS. Upon receiving 'n' numbers of requests, the UUP(e) creates a group of 'n' users. Afterward, each user forwards the query of another user in the group to the WSE. Each user collects the result for the query he has forwarded to the WSE and broadcasts the result in the group. Once all users have forwarded the single query of another member, the group finishes. To send another query, the whole process is repeated.

  In this work, we have compared the privacy achieved by proposed OSLo with the latest version of UUP(e) (Distributed system for private web search with an untrusted partner [7] proposed in 2014). In the latest version [7], queries are shuffled through optimized Bens network, and the confidentiality of the query contents are achieved through ElGamal shared key encryption. A Java-based simulator is developed to simulate UUP(e) to compute the profile privacy of a user over a privacy metric PEL. A user simulating UUP(e) only forwards a query of other group users, i.e., a self-query submission is not allowed. To have a fair comparison of profile privacy, the OSLo is simulated for a situation where a $SQFC$ only forwards the queries of other group users. The simulation result of both protocols (UUP(e) and OSLo) are collected with Dataset 1 and dataset 2. Figure 3.12 and figure 3.13 shows the profile privacy achieved by a user executing of UUP(e).

  (a) UUP(e) PEL with Dataset 1:

  Figure 3.12 represents the average PEL a simulation result of

TABLE 3.12: Average PEL of OSLo VS. UUP(e), self query-submission not allowed for Dataset 2

Number of Users	Protocol	Degree 1	Degree 2	Degree 3	Degree 4
3 users	UUP(e)	51.85	13.382	7.30	7.22
	OSLO	47.70	12.79	6.95	7.16
4 users	UUP(e)	51.156	13.14	7.08	7.19
	OSLO	48.56	12.77	6.85	6.95
5 users	UUP(e)	51.55	13.47	7.36	7.25
	OSLO	49.18	12.86	6.98	6.85

UUP(e) protocol for a group size of 3, 4, and 5 users at degree 1 to degree 4 of the ODP hierarchy. The results shown in Figure 3.12 validate the average PEL for a group size of 3 users at the first degree of the ODP hierarchy is 61.37%. The average PEL value drops to 60.88% for a group size of 4 and 57.16% for a group size of 5 users. The PEL at the second degree is 18.30%, 17.22% and 17.17% for a group size of 3 users, 4 users and 5 users respectively. The PEL for at third and fourth degree is less than 10% for all group sizes.

(b) UUP(e) PEL with Dataset 2:

Figure 3.13 shows the simulation result of UUP when simulated with Dataset 2 for the same group sizes at degree 1 to degree 4 of the ODP hierarchy. The results depict 51.85% of average PEL when 3 users are grouped together. Similarly, 51.156% and 51.55% for a group size of 4 and 5 users. However, at the second degree, a user profile for the group size of 3, 4 and 5 users is 15%, 13.13%, and 13.47% is exposed. The PEL at higher degrees (third and four degrees) of all group sizes are less than 10%.

(c) Profile Privacy Comparison of OSLo VS UUP(e)

The user executing UUP(e) forwards only a single query when the group is established, after that the group is dissolved, and to send another query the process of group creation and shuffling repeats. However, a user executing OSLo can forwards '$n-1$' queries to the WSE once the group is created. In this experiment, both protocols are simulated for a single dynamic group over Dataset 1

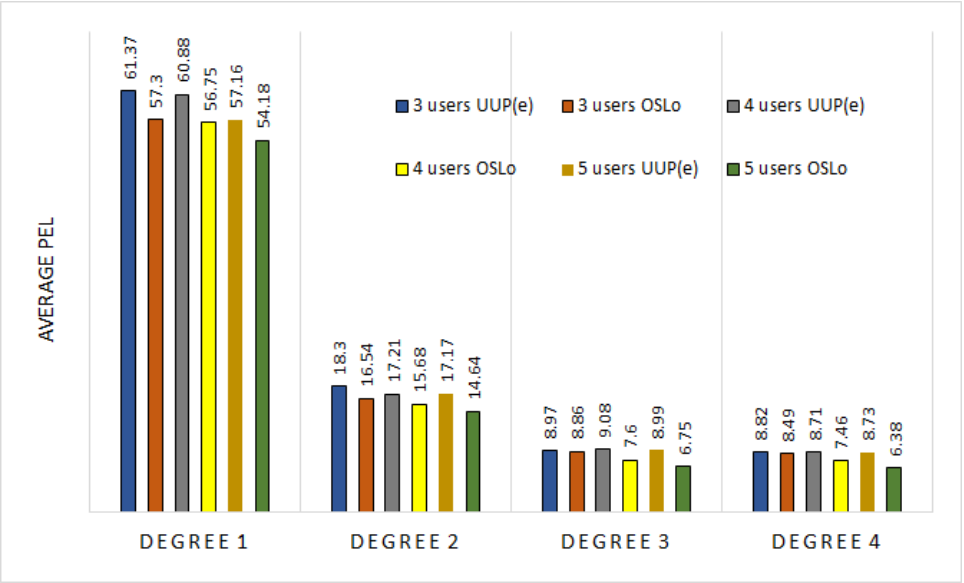

FIGURE 3.12: Avearage PEL of OSLo VS. UUP(e) for Dataset 1

and dataset 2. Considering Dataset 1, the results indicate that the user achieves better profile privacy with OSLo for all degrees of the ODP hierarchy. Results shows that for the group size of 3 users OSLo has 6.67% better profile privacy as compared to UUP(e) at degree 1 of the ODP hierarchy. Similarly, for the group size of 4 users OSLo preserved 6.89% better privacy at the degree 1 of the ODP hierarchy and 5.26% for the group size of 5 users as compared to UUP(e). Furthermore, at degree 2 of the ODP hierarchy the OSLo has 9.7% better privacy for the group size of 3 users, 9% for the group size of 4 users and 14.73% for the group size of 5 users. Likewise, at a higher degree the OSLo has better profile privacy for any group size.

Figure 3.13 and Table 3.12 shows the profile privacy comparison of UUP(e) and OSLo when simulated with dataset 2. The average PEL of a user executing UUP(e) at the first degree for the group size of 3 is 51.85% whereas, the OSLo PEL is 47.70%; similarly the PEL of UUP(e) for the group size of 4 users is 51.156% and OSLo has 48.56% at first degree. Likewise, the PEL for UUP(e) is 51.55% and OSLo has 49.18% for the group size of 5 users. Based on these results, OSLo has 8.01% better profile privacy at degree 1 of the ODP hierarchy for the group size of 3 users, 5.09% better

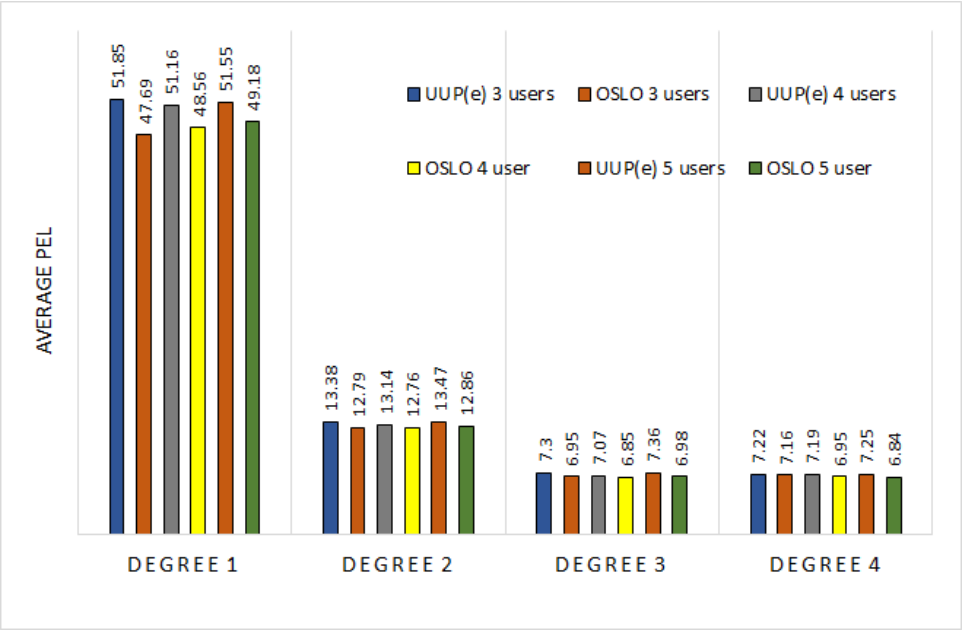

FIGURE 3.13: Avearage PEL of OSLo VS. UUP(e) for Dataset 2

profile privacy for the group size of 4 users and 4.60% for the group size of 5 users at degree 1 of the ODP hierarchy. The UUP(e) has higher profile exposure for all group sizes at all four degrees as compared to the OSLo. Furthermore, it is observed from the results that OSLo depicted an unanticipated increase in average PEL when the group size is increased to 4 and 5. Unlike, the previous situation when a self-query submission was allowed, the OSLo showed a slight increase in average PEL for the group size of 4 and 5 users. The random group is the possible reason for this increase, a user possibly grouped with those users having similar interests.

3. Profile privacy: Self-Query Submission Allowed VS self-Query Submission not Allowed:

   It is concluded from the results that a user achieve better privacy and low profile exposure for a situation when a self-query submission is not allowed as compared to allowing the self-query submission. In the latter case, a user forwards one of his/her query to the WSE when selected as $SQFC$, e.g. when there are 3 users in a group, a user forwards 33% of his/her queries. For a group of 4 users, a user forwards 25% of self queries, for a group of 5 users

20% of self queries are forwarded to WSE, and so on. However, when a self-query submission is not allowed, a user does not forward any of self query, all queries forwarded by SQFC belongs to the group users. As discussed earlier, PEL measures the difference between the user original profile and obfuscated profile, allowing the self-query submission a user has to forward a good number of self queries resulting higher PEL as compared to not allowing self-query submission. However, considering when self-query submission is not allowed, when an adversary wants to link a query to a user will exclude a user (SQFC) from the list of suspected originator, hence, narrowing the list of potential originator. Whereas, when self-query submission is allowed, an adversary cannot exclude a user (SQFC) from the list of all potential query originator users.

3.7.2 Time delay of OSLo

In any distributed protocol, a group of users collaborates to forward each other queries to the WSE. The Web search privacy achieved through any distributed protocol causes a time delay in retrieving query results from the WSE. Like other distributed protocols, privacy at the cost of delay is also a nature of OSLo. The time delay caused to achieve privacy is measured in two dimensions. i.e. time to create a group and query response delay. In this work, two experiments are performed to measure the time delay caused by the group creation process and query sending to WSE and result processing by the OSLo. A java based simulator is developed to execute OSLo using multi threads socket programming to create group, CryptoUtil library, and keypair generator methods are used to create RSA public-private key pair for query and result encryption. The experiment is performed over Intel(R) Core(TM) i3-231M CPU having 8192MB RAM over Windows 8.1 Pro 64bits. Table 3.13 shows the details of simulation equipments used to execute the OSLo.

1. Time to create a group: In OSLo, ‘n’ users collaborate to forward each other queries. The time required to create a group of ‘n’ users is one of the critical point in OSLo. This time starts when CS receives the first request from the

TABLE 3.13: Simulation details under controlled environment: equipments

Computer	CPU	Intel(R) Core(TM) i3-231M
	RAM	8 GBytes
	O.S.	Microsoft Windows 8.1 pro 64-bit
	Java version	Java(TM) SE Runtime Environment (1.8.0_65)
Network	4Mb internet connection	Java Multi-threading

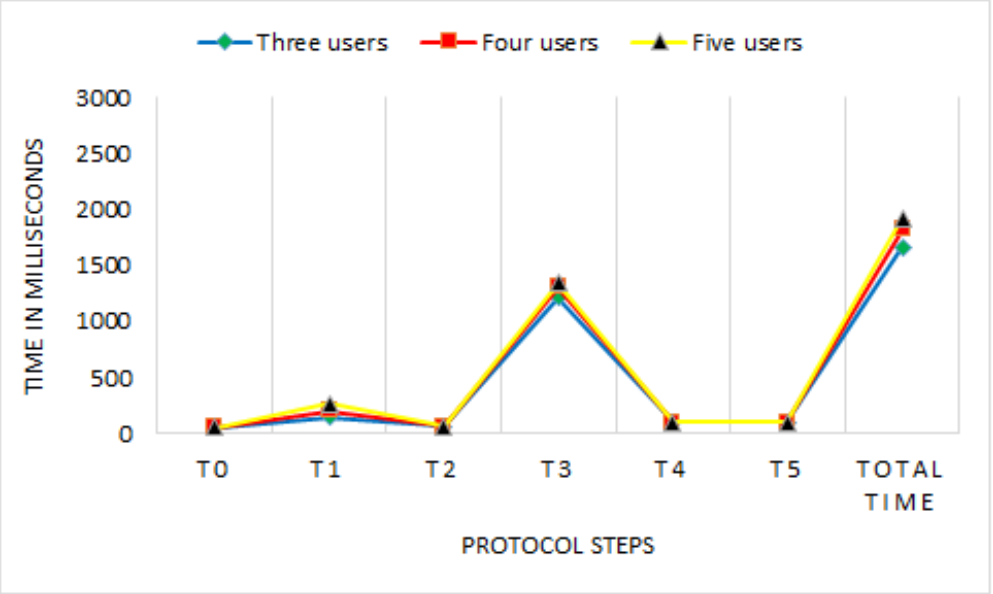

FIGURE 3.14: Time required to create a group

client and ends when the CS broadcasts the information about the group including the SQFC details and other group peers. The list contains the description of each time interval.

T0: The time required by user Ui to connect to CS, this includes CS records Ui's IP address and port number.

T1: Time required by 'n' users to connect to CS.

T2: Time required by CS to choose one user as SQFC, and forward request to share public key.

T3: Time required by SQFC to generate a public key and private key pair

T4: Time required by the SQFC to share the public key with CS this includes network delay.

T5: The time required for CS to broadcast the group information.

This experiment computed the time required to create a group of three users, four users, and five users. The experiment is performed ten times, the average time interval taken each step is shown in Figure 3.14. The result

shows that the average time required to create a group of three users is 1663ms. Similarly, a time interval of 1815ms is required to create a group size of four users and 1914ms for the group size of five users.

2. Query response delay: The objective of this experiment is to compute a delay caused by the execution of OSLo, i.e. to forward a query to WSE and retrieve the answer. The list contains a description of the time interval of each step.

   T0: Time required to make a query message, it includes generating a query 'q' of three words, generating a user U_i's encryption key(K_Ui) of 128bits and q_ID(a random number).
   T1: Time required to encrypt a query message with 1024 bit public key of SQFC.
   T2: Time required to shuffle a query among the group peers. This includes a time required to flip a coin to shuffle an encrypted query message. the network delay to pass an encrypted query from one user to another is also included in this T2.
   T3: Time required to SQFC to decrypt a query.
   T4: Time required by the SQFC to forward a query to WSE and retrieve & process results.
   T5: Time required to make an encrypted answer message (eAnsMsg) by encrypting the result file with the encryption key(K_Ui) of the user and concatenating q_ID.
   T6: Time required to broadcast eAnsMsg.
   T7: Time required to decrypted eAnsMsg by matching q_ID

The Figure 3.15 shows the time required to send a query to WSE covertly through OSLo and retrieve results. The result shows that an average time of 6446.67 milliseconds is required to send a query to WSE and retrieve results in the group of three users. Similarly, for the group of four users, the delay caused by OSLo is 6988 milliseconds whereas 6962 milliseconds for the group size of five users. It is important to note that time T4 needed the highest delay, the reason for

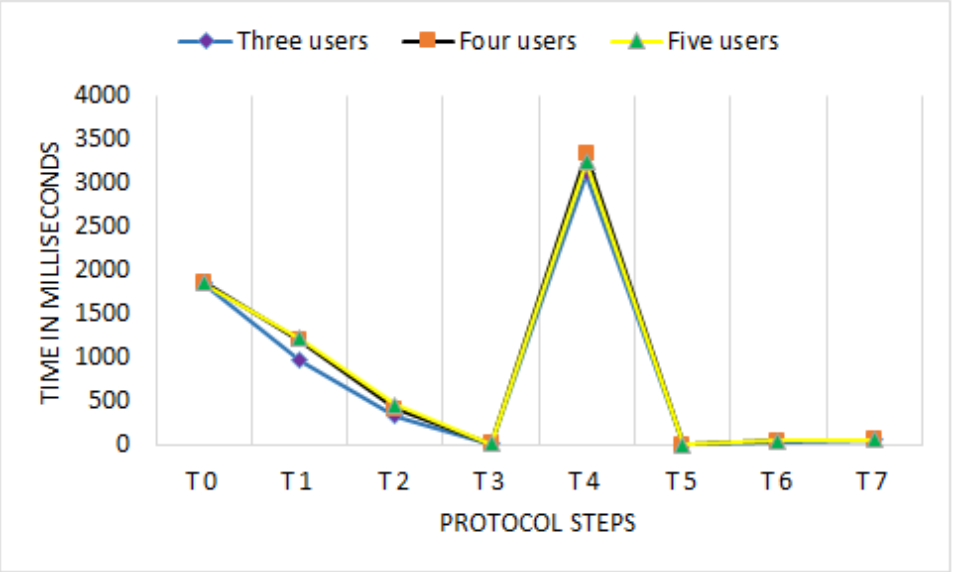

FIGURE 3.15: Time required to send a query and retrieve results

this delay is the processing of retrieved results. The first 30 results retrieved by WSE are selected and the URL of those results is written in the text files, this process incurred almost half delay of the total time. Figure 3.16 shows the delay comparison of OSLo vs other distributed protocols, the result shows OSLo cost less delay as compared to UUP(e), the reason is the UUP(e) [7] spends higher time in computing the zero knowledge proof. However, OSLo costs more time as compred to co-utile [13] and social network [43] because there were no encryption of query or result involved in the execution of these protocols. Co-utile and social network are faster but provides no local privacy and no confidentiality of query or results to the query.

3.7.3 Performance Comparison of UUP(e) vs. OSLo

The performance of OSLo and UUP(e) is analyzed on the basis of the delay caused in the group creation process and the number of groups required to execute Dataset 2. The UUP(e) protocol [7] was proposed to obfuscate the profile of a user in front of WSE. The UUP(e) executes in following steps, group setup, permutation network distribution, group key generation, anonymous query retrieval, and query submission and retrieval. In the group setup process, the CS constantly listens for the connection request. Each time a user attempts a web search privately, it sends a connection request to the CS. When the CS receives n number of requests,

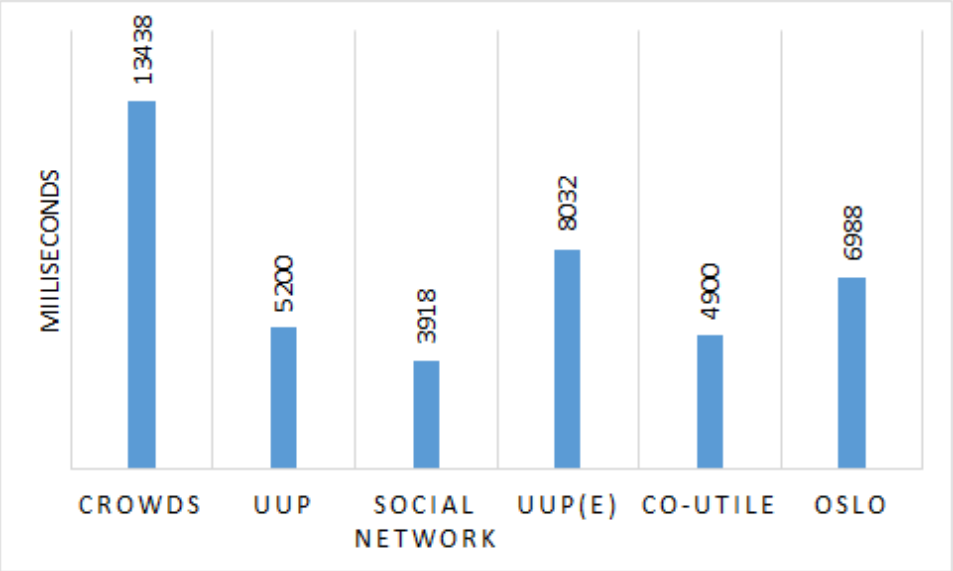

FIGURE 3.16: Delay comparison of OSLo vs other protocols

TABLE 3.14: Simulation parameters

Protocol	No of Users in a Groups	Number of Groups created	Theoretical Group Required
OSLo:Self query-submission allowed	3	11485	11516
	4	6452	6477
	5	4126	4145
	7	2093	2115
	10	1030	1036
	15	457	460
OSLo: Self query-submission notallowed	3	17259	17274
	4	11470	11516
	5	8603	8637
UUP(e): Self query-submission notallowed	3	34522	34548
	4	25858	25911
	5	20656	20728

it creates a group and informs each user about the details of other users in the group. Each user in the UUP(e) sends a single query of other users in the group to the WSE and broadcasts the results. Similarly, another user in the group forwards his /her query to the WSE. After each user has forwarded the query, the group session of UUP(e) ends. To send another query, the whole process repeats again. The UUP(e) is simulated with the group size of 3, 4, and 5 users. The working of OSLo is given in Section 3.3, The connection setup explains the procedure of a user connecting to the CS as well as the method of group creation. The group creation process of UUP(e) and OSLo is the same, both protocols follow the same steps to create the group. The key difference of OSLo is the ability to send multiple queries

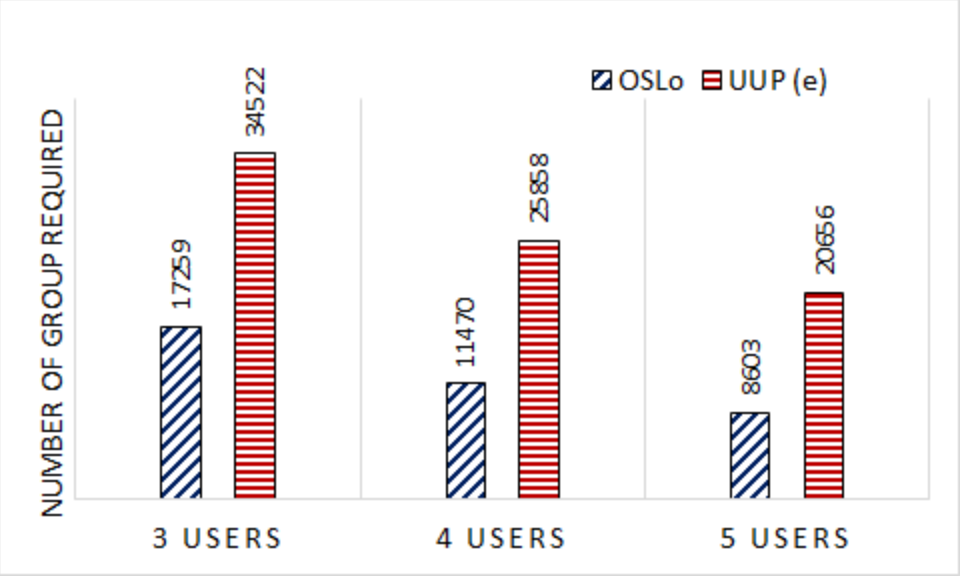

FIGURE 3.17: Number of groups required to simulate dataset 2

of the group in a single session. Once a group session is established, a user can send n queries when the self-query submission is allowed or $n-1$ queries when the self-query submission is not allowed (where n is the size of the group). The user executing OSLo requires a smaller number of groups to send the same number of queries. Table 3.14 shows the number of groups created during the simulation of OSLo and UUP(e) to send the queries mentioned in Table 3.14. Figure 3.17 shows the number of groups required to simulate dataset 2. For the group size of 3 users, UUP(e) builds 34522 groups to send the queries mention in Table 3.14. However, the OSLo made 17259 groups when the self-query submission was not allowed and 11485 numbers of groups when the self-submission was allowed. Similarly, for the group size of 4 users and 5 users, the UUP(e) made 25858 and 20646 groups to send the queries of Dataset 2, whereas OSLo required 11470 and 8603 groups. Table 3.14 shows the number of an actual group created and the theoretical required number of groups for the execution of Dataset 2. When the self query-submission was allowed the number of groups created by OSLo for simulating the Dataset 2 decreased with the group count. It is important to mention that the difference between the last two columns of Table 3.14 is due to the unequal number of queries of each user. At the end of the simulation, the protocols are left one user short of the group size required to create the group. The group creation process of a distributed protocol is a critical point, a user must wait for the $n-1$ of other users to send a connection request to the CS for the group to be created. As illustrated

[7, 40] the Poisson distribution can be used to model the number of queries a WSE receive. As mentioned in [84] the average number of queries Google answered in one second is 3996.91. If there are an average of 39.96 queries per hundredth of a second, the probability of making a group of n=3 users to send 6 queries is 1. Similarly, for the group size of n=4 to n=15, the probability of making a group is 1. According to recent statistics, Google answered 40,000 queries in one second, this means in every hundredth of a second 4000 users send their queries to Google [2]. If those users send their queries through our proposed framework, the probability of making a group size of n= 3 to 15 user is 1. Similarly, duckduckgo [6] an another WSE answers around 58M queries in a day. The duckduckgo answer an average 671 in one second i.e. 33.55 queries in fiftieth of a second (50ms) the probability of making group of three users (n=3) to send 6 queries is 1. Similarly, for the group 4 users (n=4) to 6 users(n=6) the probability of sending queries is also 1.

3.8 Limitation of OSLo

Web search privacy provided by OSLo empowers a user to protect their self from identification and profiling. Although a user executing OSLo achieves both local privacy and profile privacy but the user has to compromise on a certain thing. A user may not get personalized search results, the quality of results returned by the WSE may not as accurate as a user gets with the profiling by the WSE. Like any other distributed protocol, a delay is another prime concern in OSLo. The whole execution process of OSLo causes delay as compared to the prompt response if a user had forwarded the query directly to the WSE. In OSLo, the SQFC is considered curious but honest, SQFC has to perform his or her role of forwarding queries to WSE genuinely. However, if SQFC is corrupt and does not forward queries of the group peers to the WSE can cause significant delay or in the worst case no query result at all. Like other distributed protocols, the ethical issue is an additional matter associated with OSLo. A user may forward a dangerous query on behalf of group peers may have a grave outcome, and a user has to face the consequences for such queries.

[6] https://duckduckgo.com/traffic

3.9 Conclusion

This chapter presents a novel distributed privacy-preserving protocol ObScure Logging (OSLo) to tackle the limitation mentioned in the initial paragraph of this chapter and to provide better privacy and performance. The first limitation mentioned is that the user associated with a common memory location can see the query content and the result to the query once it is written in the memory location. In OSLo, the user encrypts the query with the public key of $SQFC$, so no user in the group can see the query contents. Similarly, the second limitation is that results retrieved from WSE are broadcasted in clear text giving the group user an idea of what is being searched by an individual in the group's existing protocols. The $SQFC$ encrypts the results retrieved from WSE with the encryption key of the query originating user, hiding the results from the group users. The self-query submission was not allowed in the existing protocol, which gives an attacker a clue to exclude a query-forwarding user from the anonymity set, hence narrowing the anonymity set. However, the OSLo allows the self-query submission to eliminate this limitation. The co-utile protocol offered no privacy relative to the peer users, but OSLo preserves the local privacy through shuffling and encryption. The performance of the OSLo is compared to the UUP(e) based on the number of the groups required to send the same number of queries to the WSE. The OSLo requires far fewer groups as compared to UUP(e) when forwarding the same number of queries.

This chapter answer the research question 1 of dissertation mentioned in Section 1.8. **RQ 1.** *How to improve the local privacy and profile privacy of a user in a private web search?"*

- RQ 1 (a) *What will be the effect of allowing self-query submission and not allowing self-query submission on profile privacy of the user?*
- RQ 1(b) *How does the* ***size*** *of the group and group count affect the privacy and performance of the protocol?*

In this chapter, the proposed framework ObScure Logging (OSLo) tackles the RQ1. The OSLo provides both local privacy and profile privacy. The local privacy is

achieved through query encryption, query shuffling, result encryption, and broadcasting, whereas, the profile privacy is attained through forwarding the queries of other users. The query encryption makes the query contents hidden from the group users; the query shuffling breaks the link between the query and the originating user. The $SQFC$, when receiving a query from the user will not be used if the user who has forwarded a query to him is the query originator or just the query forwarder. The results are encrypted with the encryption key (asymmetric key) provided by query originating user makes the query contents hidden from group users, also the result broadcasting is the final step to achieve the local privacy. The OSLo achieves profile by obfuscating the profile of a user with the queries of other group users. This obfuscating reduces the risk of the WSE classifying a query as a machine-generated query. Every time the user is grouped with other users having a range of interests, hence the profile of a user has obfuscated the queries of users who have a variety of interest, resulting in a significant obfuscation of user profile.

1. "How the size of the group affects the privacy and performance of the protocol?"

   The Equation 3.4 and Equation 3.5 show that the local privacy is depending on the size of the group. When the number of users 'n' increases, the probability of linking the query to the originator decreases. Similarly, the profile privacy also affects the size of the group as shown in Table 3.5 - Table 3.8. In this chapter, the performance of OSLo is compared with UUP(e) on the basis of the number of those groups required to send the number of queries in dataset 2. The results mentioned in Table 3.3 shows OSLo requires a fewer number of groups to send the same number of queries as compared to the UUP(e), hence OSLo offers better performance as compared to UUP(e).

2. "What will be the effect of allowing self-query submission and not allowing self-query submission on profile privacy of the user"?

   Section 3.7.1 provides the detailed description of the effects for allowing self-query submission and not allowing self-query submission on the profile privacy of a user. The result of not allowing the self-query submission is compared with the state-of-the-art privacy preserving protocol UUP(e). The

results show that OSLo improves the profile privacy of a user as compared to UUP(e) with simulated with the selected datasets.

Chapter 4

Multi-Group ObScure Logging (MG-OSLo)

The previous chapter discussed the single group distributed protocol to preserve the web search privacy of a user. Domingo-Ferrer and Bras-Amor´ [39] introduced the concept of multi-group to preserve the privacy of a user. UPIR was implemented with the use of drop boxes or memory location, where a subset of users was associated with each memory location. Over time, there are a few variations of UPIR presented, such as the one-to-one, all to all UPIR protocols and configuration-based protocol. Descriptions of these protocols are presented in Section 2.5.2. Later the concept of UPIR is extended by authors in [41, 42, 67, 85]. Swanson and Stinson presented an extended combinatorial construction for peer-to-peer UPIR protocol by strengthening the existing UPIR protocol, provided privacy analysis of UPIR protocols using Balance Incomplete Block Design (BIBD) and introduced an attack called intersection attack [41]. Later, Swanson and Stinson extended their work by providing extended results for UPIR protocols [42]. However, the entire existing variant evaluated the privacy of users only relative to the peer users (local privacy).

The following are the drawbacks of the existing multi-group distributed protocols:

i. any user connected with the memory location can see the query contents.

ii. the result of the query is written back in the same memory location, so the

user associated with the memory location can see the contents of the result.
iii. these schemes are vulnerable to intersection attack [39, 41, 67, 85].

To the best of our knowledge, the privacy of a user in multi-group protocols is only evaluated relative to the group users involved in forwarding query to the WSE. The existing multi-group distributed protocols evaluate the privacy of a user in only one dimension i.e., the local privacy. The profile privacy (the privacy of users relative to the WSE) in multi-group distributed protocol has never been evaluated. This chapter proposes, a multi-group distributed privacy-preserving protocol MG-OSLo (Multi Group ObScure Logging) that preserves and evaluates the local privacy and profile privacy of a user. The privacy of a user achieved through the MG-OSLo is evaluated in two dimensions, the local privacy computes the probability of linking a query by a dishonest user with the query originating users. Whereas, the profile privacy computes the impact of multi grouping on the magnitude of profile obfuscation. The MG-OSLo consists of multiple groups, each group accommodates a set of users to perform a web search secretly. MG-OSLo does not use the concept of the memory location, instead, users are grouped by a Core server (CS). The Section below presents the description of MG-OSLo.
The objectives of MG-OSLo include:

1. To devise a mechanism a Multi Group distributed privacy-preserving protocol (MG-OSLo) that preserve the local privacy and profile privacy of a user in private web search.

2. To evaluate the privacy preserved by executing MG-OSLo in two dimensions i.e., the impact of multi grouping on local privacy and profile privacy.

3. To analyze the grouping of users using various approaches such as non-overlapping design, overlapping design, and BIBD

4. To analyze analytically the local privacy of a peer for BIBD (v, b, r, k, λ) configuration. A probabilistic advantage to an entity of MG-OSLo in linking query with the user in grouping design.

5. To provide empirical evaluation to measure the privacy of a user relative to WSE. An experiment is performed to calculate the magnitude of profile

obfuscation for multiple group count over multiple datasets using a privacy metric PEL.

6. To analyze the impact of group size and group count on local privacy and privacy relative to the profiling of WSE.

Definition: Non-overlapping group design: In this design, each user is associated with a single group. All groups are distinct, but a user can send a query through the users associated with other groups.
Definition: Overlapping group design: Each user is associated with multiple groups simultaneously There are different grouping design approaches available, however, in this work, BIBD is used to group the users together.

4.1 Multi-Group ObScure Logging (MG-OSLo)

In this chapter, a multi-group distributed privacy-preserving protocol MG-OSLo is proposed which consists of 'b' number of groups and each group has 'k' number of the user. The section below explains the entities and execution process of MG-OSLo. The entities and working of the MG-OSLo are explained below.

4.1.1 Entities

Following are the entities required for the executing of MG-OSLo.

1. User: An individual who intends to search a query over the WSE covertly.

2. Group Search query-forwarding client ($GSQFC$): A user selected by CS to forward queries of group users to WSE for the specified duration. Each group will have GSQFC, which is supposed to forward queries to WSE. The GSQFC takes part in query shuffling.

3. Core server (CS): A dedicated machine that supervises the working of the protocol. The CS is responsible for group creation, selection of GSQFC and CS does take part in query shuffling. In the proposed MG-OSLo, the CS is

considered an honest but curious machine that performs its duties honestly and accordingly.

4. Web Search Engine (WSE): A software system, that searches data in the internet based on keywords.

4.1.2 MG-OSLo Execution Process

The Core Server (CS) continuously listens to the connection request from the users. When the CS receives a connection request, it enqueues a user into a group having a vacant slot. The CS creates a new group when a group has reached its maximum size. The CS selects a user as GSQFC for each group who is supposed to forward K queries to the WSE. Once the GSQFC has forwarded the desired number of queries the next user in the group is selected as GSQFC. Every user is selected as a GSQFC in round robin fashion. GSQFC also takes a part in query shuffling, the process of shuffling is explained in the latter Section. It is important to mention that if a user has 'X' number of queries to be forwarded to WSE, he forwards 'X' queries of the group users to the WSE and his or her queries are be forwarded by other users when selected as GSQFC. Once the desired group count and group size is complete, and GSQFC is designated for each group the CS then broadcasts the information about groups, users in each group, and the GSQFC of each group. Each time the GSQFC has changed the CS updates this information accordingly.

To send a query covertly through MG-OSLo, a user generates a query 'q', encryption key(K_Ui) and a random number (q_ID). The user then makes a query message (QMsg) by concatenating these three strings. In the next step the user randomly select a GSQFC and encrypts the QMsg with the encryption key (public key) of that GSQFC creating an eQ, in the following step the users appends a G_ID and makes an eQ_Msg. After the completion of query encryption process, the user shuffles the eQ_Msg in two-level, i.e., intra-group shuffling and inter-group shuffling. The shuffling breaks the link between the query and user (means the query become unlinkable with the user who has created it). In the intra-group shuffling, the query is shuffled among the group users, whereas, in the

inter-group shuffling the query is shuffled among GSQFC to hide the group ID of the user. When the eQ_Msg reaches the GSQFC, it decrypts the eQ_Msg and forwards the 'q' to the WSE. The GSQFC collects the query result, afterward the GSQFC encrypts the query result with the public key of the originating user creating an $eAns\_msg$. The $eAns\_msg$ is forwarded to the CS, which relays it to all GSQFCs, the group where from which the query originator from the GSQFC will broadcast it in the group. The query sending process of MG-OSLo is shown in Figure 4.2

Following are the steps required in the execution of MG-OSLo.

1. Connection setup and Group formation:

   A client-side software enables the user to connect to the CS. Once the connection is established, the CS records the user's credential (IP and port number) the CS places the user in a group having a vacant slot. If there is no vacant slot in a group, the CS creates a new group. A user receives a list of all online peers in the group from CS and the information of GSQFC. Each group will have 'K' users. Algorithm 4 line 1 to 11 shows the connection setup process. Each user in the group will be able to forward K queries to WSE through any GSQFC. The CS keeps track of users moving in and out of the group.

2. $GSQFC$ Selection

   The CS selects a GSQFC from group users in round-robin fashion for each group. Algorithm 4 line No. 13 to 20 shows the GSQFC selection process. When a user is selected as GSQFC, it generates asymmetric keys and shares a public key and G_ID with CS. The information about GSQFC encryption key and G_ID are broadcasted to all users in MG-OSLo (Algorithm 4 line No. 21). A user selected as GSQFC forwards K queries to the WSE, afterward another user is selected as GSQFC. Algorithm 7 shows working for GSQFC.

3. Query Sending Process

   Figure 4.1 depicts the activity diagram of query sending process of MG-OSLo. The query sending process consists of the following steps:

Algorithm 4 MG-OSLo: *ServerSideAlgorithm*

Input: $(Connection\_request, relieved\_Msg, eQ\_Msg, eAns\_Msg, M\ 'number\ of\ Groups', K\ 'size\ of\ group')$

Output: $(group\_information, group\_user\_list[M][K])$

1: **procedure Connection Algorithm**

2: ***Receive(Connection_request)***
3: $server \leftarrow connection\_request.accept()$
4: **for** $i \in M$ **do**
5: **for** $j \in K$ **do**
6: $G[i][j] \leftarrow enqueue(get(IP, port))$
7: $Let\ \ counter\_variable \leftarrow 0$
8: $Select\ GSQFC\big(G[\,][counter\_variable]\big)$
9: ***Receive(relieved_Msg)***
10: $counter\_variable++$
11: $Select\ GSQFC\big(G[\,][counter\_variable]\big)$
12:
13: **procedure group search query forwarding client selection**

14: ***GSQFC*$\big($*G[ ][counter_variable]*$\big)$

15: ***Select GSQFC*$\big($*G[ ][ ]*$\big)$
16: **for** $x \in M$ **do**
17: $GSQFC[\,] \leftarrow G[x][counter\_variable]$
18: **for** $y \in M$ **do**
19: $GSQFC\_list[\,] \leftarrow forward\big(GSQFC[y], get\_GSQFC\_infoMSG\big)$
20: $return(details)$
21: ***Broadcast*** $($***G[ ][ ], GSQFC_list[ ]***$)$
22:
23: **procedure : Forwarding eQ_Msg to all GSQFC**

24: ***Receive(eQ_Msg)***
25: **for** $i \in GSFQC\_list.size$ **do**
26: $Forward\big(GSQFC[i], eQMsg\big)$
27:
28: **procedure : forwarding eAns_Msg to all GSQFC**

29: ***Receive(eAns_Msg)***
30: **for** $i \in GSFQC\_list.size$ **do**
31: $Forward\big(GSQFC[i], eAns\_Msg\big)$

Algorithm 5 OSLo: Client Side Algorithm

Input: $\big(G[\,][\,], GSQFC\_list[], eAns\_Msg, get\_GSQFC\_info\_MSG\big)$

Output: $(eQ\_Msg, GSQFC\_details)$

procedure : Query sending process
    $q \leftarrow generate\_query()$
    $GSQFC \leftarrow Select\_randomly\big(GSQFC\_list[\,]\big)$
    $pbK\_GSQFC \leftarrow get\_pub\_Key\big(GSQFC\_details\big)$
    $G\_ID \leftarrow get\_ID\big(GSQFC, details\big)$
    $Query\_content\_encryption(q)$

procedure : Query_content_encryption(q)
    $Encryption()$
      $generate\_key()$
       $K\_U_i \leftarrow get\_ekey()$
      $generate(q\_ID)$
     $QMsg \leftarrow concatenate(q + K\_U_i + q\_ID)$
     $eQ \leftarrow eKP_bK\_GSQFC(QMsg)$

procedure : *eQ_Msg creation*
    Generate_eQMsg_packet()
      $eQ\_Msg \leftarrow pcf(eQ + G\_ID)$

procedure : *intra group shuffling*

    intra_group_shuffling(eQ_Msg)
      $X \leftarrow generate\_random\_number(1, 10)$
    if $x \leq 5$ **then**
       $y \leftarrow get\_random\_user\_details\big(G[g\_ID]\big)$
       $forward(eQ\_Msg, y)$
    else
       $forward(eQ\_Msg, GSQFC)$

procedure : *result decryption process*
    Receive(eAns_Msg)
      $q\_ID \leftarrow get\_id(eAnsMsg)$
    if $q\_ID.match$ **then**
         $Enres \leftarrow get\_result(eAnsMsg)$
         $result \leftarrow decrypt(Enres)$

36: **procedure :** *GSQFC details generation*

37: ***Receive(get_GSQFC_infoMSG)***
38: $get\_GSQFC\_details\_info()$
39: $Generate\_Asymmetric\_Keys()$
40: $P_bK\_GSQFC \leftarrow get\_public\_key()$
41: $P_rK\_GSQFC \leftarrow get\_private\_key()$
42: $Generate\ ID()$
43: $G\_ID \leftarrow random\_number()$
44: $port \leftarrow get\_port()$
45: $generate\_details\_msg()$
46: $details \leftarrow dMsg(P_bKGSQFC, port, G\_ID)$
$\backslash\backslash details\ message\ consisting$
47: $return(details)$

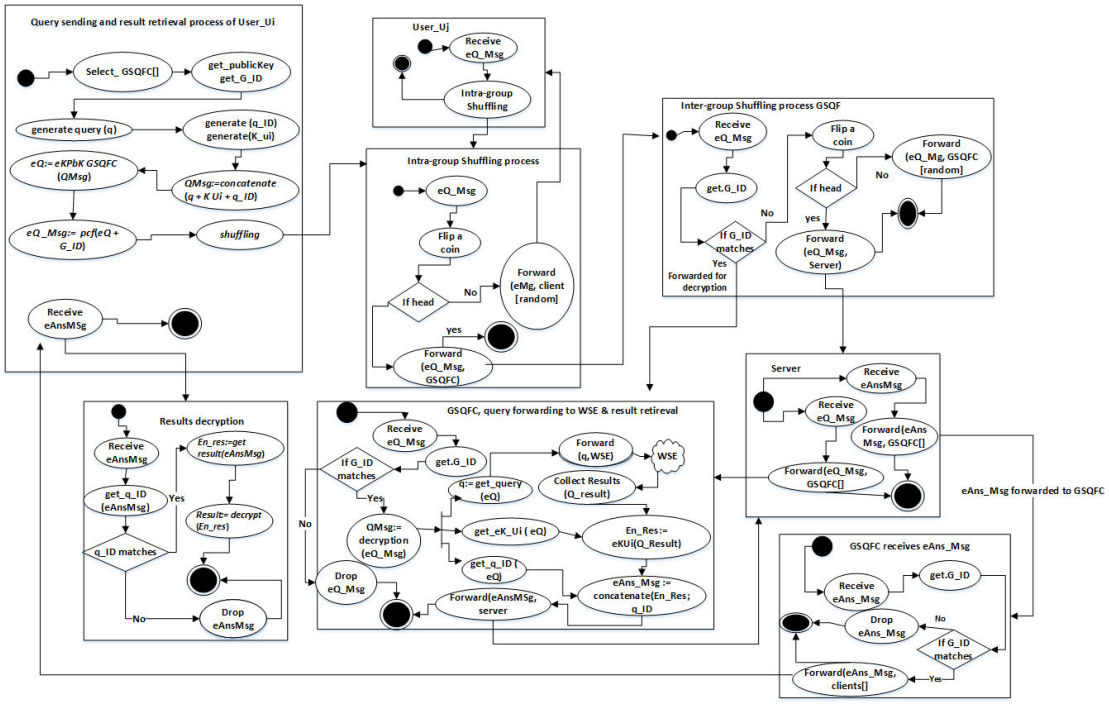

FIGURE 4.1: MG-OSLo: Activity diagram of query sending process

(a) Encryption

To send a query, the user performs the following tasks, i) generate the query 'q', generate an encryption key K_U_i and a query id (q_ID). The K_U_i is a 128bits AES share key that are be used for encrypting the query result (r) while q_ID will be used by a query originating user to recognize that the result(r) is of his query(q). The user concatenates the 'q', K_U_i and q_ID together and makes a packet called a $QMsg$ packet. In the next step, the user selects a GSQFC from the list of all GSQFCs and encrypts the QMsg with the public key of the GSQFC

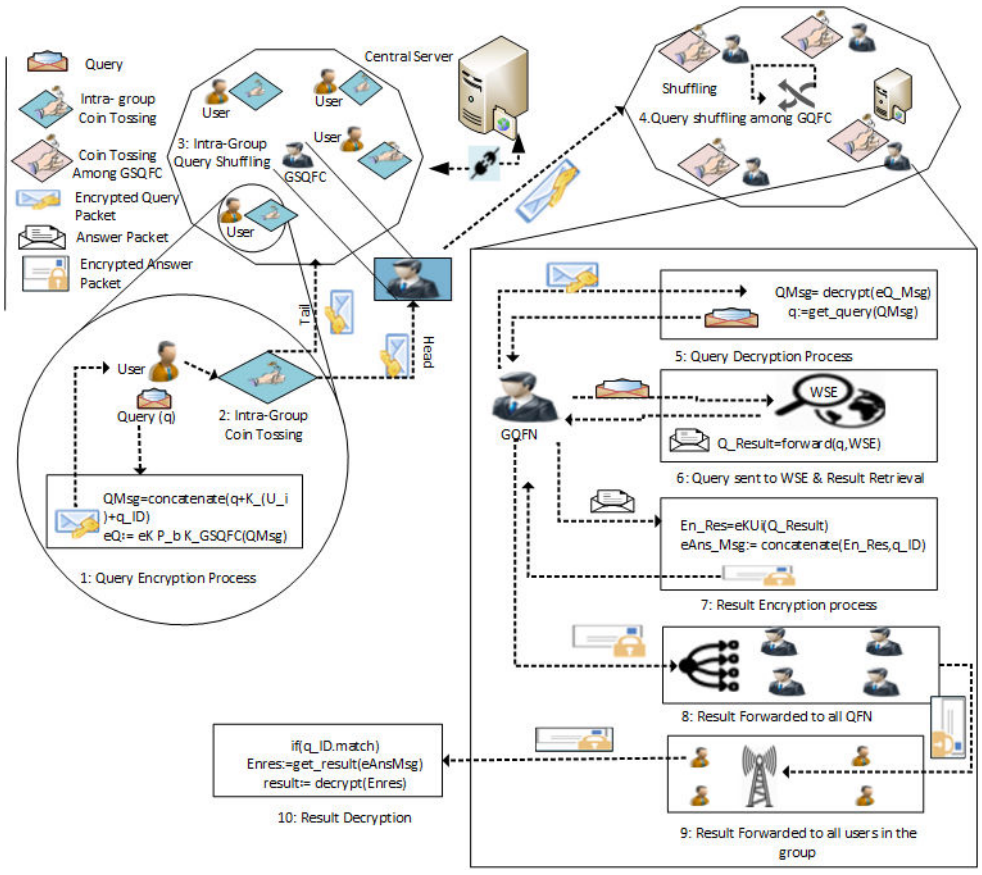

FIGURE 4.2: MG-OSLo: Graphical representation of query sending process

using a function $PbK_GSQFC(QMsg)$ and making an eQ packet. The query encryption process is shown in Algorithm 5 Line No. 8 - 14. The user then appends G_ID with eQ making an encrypted query message eQ_Msg by using a packet creation function. The G_ID will be used by GSQFC to confirm that query (q) is encrypted with his public key and he is supposed to forward the query (q) to WSE. The process of encryption is shown in Algorithm 2 line 17 - 18.

(b) Query Shuffling

Encrypted query message (eQ_msg) is shuffled in two stages i.e., intra-group shuffling and inter-group shuffling.

- Intra-Group Shuffling

  To shuffle the eQ_Msg among the group peers, a coin is tossed by the user to decide the destination of eQ_Msg. If the coin lands on head, the eQ_Msg are forwarded to GSQFC and intra-group shuffling ends there. However, if the coin lands on tail, the eQ_Msg is forwarded to another group peer selected randomly

from the list of all peers. The intra-group shuffling is shown in Algorithm 5 line 20 - 27.

- Inter-Group Shuffling

  This level of shuffling is used to hide the group identity of the user. When GSQFC receives an eQ_Msg, it will try to decrypt in order to check the G_ID whether the query(q) is encrypted with their public key, otherwise GSQFC will toss a coin to decide the destination for the eQ_Msg. If the toss results in tail, the eQ_Msg will be forwarded to another GSQFC selected randomly from the list of all GSQFCs. The GSQFC will follow the same steps mentioned above, else, if the toss results in head, the eQ_Msg is forwarded to CS. Algorithm 6 shows the process of Inter group shuffling. The CS forwards the eQ_msg to all $GSQFC$ as shown Algorithm 4 line 23 - 26. The $GSQFC$ will check the G_ID to check if the query is encrypted with his public key, Algorithm 7 line 4.

Algorithm 6 M_G-OSLo: Inter group shuffling

Input: (GSQFC_list[ ], eQ_Msg)

Output: (*query shuffled before reaching the server*)

```
1: procedure : inter_group_shuffling
2:     Receive(eQ_Msg)
3:     x ← generate_random_number(1, 10)
4:     if x ≤ 5 then
5:         y ← get_random_GSQFC[ ])
6:         forward(eQ_Msg, y)
7:     else
8:         forward(eQ_Msg, server)
```

4. Query forwarding to WSE and Result Retrieval:

   The eQ_msg is decrypted using decrypt(eQ_msg) function by the GSQFC. The query(q) is forwarded to the WSE. The q is processed by WSE and the results(Q_Result) are returned to the GSQFC. The GSQFC encrypts the result(Q_Result) with the encryption key of the user (eKUi) making En_Res. The En_Res are appended to the q_ID, thus creating an encrypted answer message $eAns\_Msg$. The GSQFC forwards $eAns\_Msg$ to

CS, which forwards the results to all GSQFCs. The GSQFC broadcasts the $eAns\_Msg$ to all peers in the group. The user who has the decryption key will decrypt the required result. The q_ID is used as an identifier to confirm if the results are for the queries what originator has sent. Algorithm 7 shows the query forwarding and result retrieval process.

The eQ_msg are decrypted using decrypt(eQ_msg) function by GSQFC. The query(q) is forwarded to WSE. The q is processed by WSE and the results(Q_Result) are returned to the $GSQFC$. The $GSQFC$ encrypts the result(Q_Result) with the encryption key of the user (K_U_i) making En_Res. The En_Res are appended to the q_ID, thus creating an encrypted answer message $eAns\_Msg$. The $GSQFC$ then forwards $eAns\_Msg$ to CS, which forwards the results to all $GSQFC$s. Algorithm 7 line No. 1 - 13 shows the query sending and result retrieval process, The $GSQFC$ broadcasts the $eAns\_Msg$ to all peers in the group, as shown in algorithm 7 line No. 20 - 23. The user when receives the $eAns\_Msg$ will try to decrypt the packet, however, the user who has the decryption key will decrypt the required result. The q_ID is used as an identifier to confirm if the results are for the queries what originator has sent. Algorithm 5 line No. 29 - 34 shows the result decryption process.

5. Search Query Forwarding Client selection

   The CS selects $GSQFC$ for each group from the list of users in that group on first come first serve basis. Each $GSQFC$ is supposed to forward K queries to the WSE, once $GSQFC$ has forwarded K queries, next user will be selected as $GSQFC$ (where K is the number of users in a group). $GSQFC$ is supposed to do the following task.

   (a) It generates a public key and a random number (G_ID) and gives information to CS. The CS broadcasts $GSQFC$'s public key information in all groups. The random number is used for identification by $GSQFC$ to confirm that the query is encrypted under his/her public key.

   (b) It sends a query to WSE and downloads the answer to the query. Creates an encrypted answer packet after encrypting query with the encryption key of the user, and broadcast it to all other $GSQFC$.

Algorithm 7 M_G-OSLo: Group source query forwarding client

Input: (eQ_Msg,eAns_Msg,threshold)

Output: ($forward\ msg\ to\ server$, $broadcast\ msg\ in\ group$, $forward\ msg\ to\ WSE$)

1: **procedure : query forwarding and result retrieval from WSE**
2:
3: ***Receive(eQ_Msg)***
4: $get.G\_ID(eQMsg)$
5: **if** $G\_ID.match()$ **then**
6: $QMsg \leftarrow decrypt(eQMsg)$
7: $q \leftarrow get\_query(QMsg)$
8: $K\_U_i \leftarrow get\_usrkey(QMsg)$
9: $q\_ID \leftarrow get\_q\_ID(QMsg)$
10: $Q\_Result \leftarrow forward(q, WSE)$
11: $c\_variable++$
12: $En\_Res \leftarrow eK\_U_i(Q\_Result)$
13: $eAns\_Msg \leftarrow concatenate(En\_Res, q\_ID)$
14: $forward(eAns\_Msg, server)$
15:
16: **if** $c\_variable.equal(threshold)$ **then**
17: $forward(server, relievedMsg)$
18: **else**
19: $Inter\_group\_shuffling()$
20:
21: **procedure : received encrypted answer packet**
22: ***Receive(eAns_Msg)***
23: **for** $i \in groupsize$ **do**
24: $Forward(eAnsMsg, G[G\_ID][i])$

(c) *GSQFC* takes part in intra group query shuffling. This level of shuffling ensures unlinkability of group ID. This will not allow the adversary to link a query with any group or user.

(d) When *GSQFC* receives a query packet from a group peer user, it attaches a hash with a query packet for answer packet identification.

4.2 Privacy Evaluation of MG-OSLo

As mentioned in Section 3.1, the privacy of a user in the distributed privacy-preserving scheme is considered preserved if the user achieves to preserve local

privacy and profile privacy. They local privacy is attained by encryption (query content and query result encryption) and shuffling. The encryption makes the query and results unreadable while shuffling breaks the link between the user and the query making the query and user unlinkable. Profile privacy is achieved by polluting the profile of the user with the queries of group users. The privacy of a user executing MG-OSLo is evaluated in two dimensions, i.e., local privacy and profile privacy. The local privacy measures the unlinkability whereas, the profile privacy quantifies the indistinguishability. The local privacy measures the probabilistic advantage a curious user has in linking a query with the originating user, whereas the profile privacy computes the level of profile obfuscation. The section below gives a comprehensive evaluation of local privacy and profile privacy.

4.2.1 Local Privacy

The local privacy of the user depends on the way the users are grouped. The users can be grouped in different ways, each grouping method affects the privacy of a user. MG-OSLo considers two grouping designs i.e. A) Non-overlapping group design. B) Overlapping group design.

A Non-overlapping group design

In this design, each user appears in a single group. Each group has its own $GSQFC$, a user can send a query through any $GSQFC$.

v total number of users i.e., $U = \{U_1, U_2, U_v\}$

b total number of groups i.e., $M = \{S_1, S_2, ..S_b\}$

r degree of a user, i.e., how many groups a user is associated with. In non-overlapping design, r is one because each user appears in a single group.

k is the number of users in a single group.

We have considered three random variables, S, M and P. where S represents a source of a query, M denotes a group, and P indicates proxy ($GSQFC$ or CS) who forwards the query to $GSQFC$.

The subsection below describes the local privacy of a user relative to the entities of MG-OSLo. When non-overlapping group design is considered for

grouping users and a curious entity of MG-OSLo wants to link a query with the user, the probability of linking query is given as:

(a) Privacy Relative to $GSQFC$

Considering the working of MG-OSLo, when a $GSQFC$ receives a suspicious query and interested to find the query source, what advantages does a GSQFC have in linking the query with the user. According to the shuffling technique of MG-OSLo, the query is shuffled before reaching $GSQFC$. The probability of linking query with the user.

$$Pr(S = U_i, M = S_l \,|P = GSQFC) = Pr(M = S_l \mid P = GSQFC) \cdot Pr(S = U_i \mid M = S_l, P = GSQFC) \tag{4.1}$$

$$Pr(M = S_l \;\mid\; P = GSQFC) = \frac{1}{(b-1)} \tag{4.2}$$

where, b is the total number of block (groups). $GSQFC$ knows that query does not belong to their group so he excludes their group.

$$Pr(S = U_i \mid M = S_l, P = GSQFC) = \frac{1}{K} \tag{4.3}$$

Where k is the total number of users in each block.

Equating 4.2 and 4.3

$$Pr(S = U_i, M = S_l \mid P = GSQFC) = \frac{1}{(K(b-1))} \tag{4.4}$$

If $GSQFC$ wants to link query with the user, the probability depends on a number of users and groups, shown in (4.4). $GSQFC$ does not get any advantage in linking query with the user except excluding their group.

(b) Privacy Relative to the Core Server

The CS receives the encrypted query after the shuffling process finishes. The user achieves data confidentiality due to the query contents and the results are encryption. As a result, the CS will not be able to discover the content of a query. If GSQFC or any other user does not collaborate with CS, the probability of linking the query to the user is

$\frac{1}{v}$, because all users are equally probable, where 'v is the total number of users in all groups. According MG-OSLo, CS gets no advantage in linking the query with the user. However, if the GSQFC and CS form a coalition to find the query source, the probability of linking the query to the user is given by:

$$\begin{aligned} &Pr(S = U_i, M = S_l \mid P = GSQFC, S \notin M_c) = \\ &Pr(M = b_i \mid P = GSQFC, S \notin M_c) \\ &\cdot Pr(S = U_i \mid M = S_l, P = GSQFC, S \notin M_c) \end{aligned} \tag{4.5}$$

M_c is a group whose $GSQFC$ has made a coalition with CS.

$$Pr(M = S_l \mid P = GSQFC, S \notin M_c) = \frac{1}{(b-1)} \tag{4.6}$$

$$Pr(S = U_i \mid M = b_i, P = GSQFC, S \notin M_c) = \frac{1}{K} \tag{4.7}$$

Equating 4.6 and 4.7

$$Pr(S = U_i, M = S_l \mid P = GSQFC, S \notin M_c) = \frac{1}{(K(b-1))} \tag{4.8}$$

The CS can only exclude the group that GSQFC has made a coalition with the CS shown in (4.8); all other groups and users are equally probable if C GSQFC made a coalition with CS, all of those groups would be excluded from the anonymity list and the probability of link query would be $\frac{1}{(b-C)K}$.

(c) Privacy Relative to Group Users:

As discussed earlier, the query and the result contents are encrypted under the public key of the GSQFC, the user achieves unlinkability and confidentiality for users in the same group or other groups. However, if all GSQFCs are compromised, i.e., where from the query has originated, the GSQFC who has sent the query to the WSE and the GSQFC of other groups including the CS, build a coalition to find the query originator. Equation (4.9) gives the probability of linking the query to the user when the above-mentioned entities form a coalition

to find the query originator.

$$Pr(S = U_i \mid M = S_l, P = GSQFC) = \frac{1}{K} \tag{4.9}$$

If C number of users collaborate with the GSQFC and the CS to find the user, the probability of linking the query to the user is presented in (4.10).

$$Pr(S = U_i \mid M = S_l, P = GSQFC) = \frac{1}{(|K| - C)} \tag{4.10}$$

B Overlapping Group Design

If an overlapping design is used, i.e., a user appears in multiple groups, one user (GSQFC) is supposed to forward the queries to WSE. The design is described as

v total number of users i.e., $U = \{U_1, U_2, ...U_v\}$

b total number of groups i.e., $M = \{S_1, S_2, ...S_b\}$

r degree of user, i.e., the user association with number of groups

k number of users in a single group

λ pair of user appearance in a group if λ =1 means, a pair of users appears in a single block, λ =2 means a pair appears in two block Definition: Balance Incomplete Block Design (BIBD): a (v, b, r, k, λ) design in which every pair of points occurs in exactly λ blocks [86]. In the MG-OSLo, users are grouped using BIBD approach to evaluate the impact overlapping design on local privacy.

Let's consider the first case where (v, b, r, k, λ) design is used, S= source, P= proxy (a user who forwards the query to $GSQFC$), M=group, Group Query Forwarding Node ($GSQFC$) user who for¬wards query to the WSE

(a) Source and $GSQFC$ Belong to Different Groups

A user appears in multiple groups and overlapping design is used. Where $GSQFC$ is supposed to forward the queries of all peer users to the WSE. $GSQFC$ and query source are not from the same group.

$$Pr(S = U_i, M = S_l \mid P = U_j, GSQFC \notin S_l) = \frac{Pr(P = U_j, GSQFC \notin S_l \mid S = U_i, M = S_l) \cdot Pr(S = U_i, M = S_l)}{Pr(P = U_j, GSQFC \notin S_l)} \tag{4.11}$$

$$Pr(S = U_i, M = S_l) = Pr(S = U_i) \cdot Pr(M = S_l \mid S = U_i) \tag{4.12}$$

$$\begin{aligned} Pr(S = U_i) &= \sum_{i=1}^{b} Pr(M = S_l) \cdot Pr(U_i \mid M = S_l) \\ &= \sum_{i=1}^{b} \frac{1}{b} \cdot \frac{1}{K} \end{aligned} \tag{4.13}$$

$$Pr(S = U_i) = \frac{r}{(b.K)} \tag{4.14}$$

$$Pr(M = S_l \mid S = U_i) = \frac{r}{b} \tag{4.15}$$

Equating 4.14 and 4.15

$$Pr(S = U_i, M = S_l) = \frac{r}{b} \cdot \frac{r}{b \cdot K} \tag{4.16}$$

$$Pr(P = U_j, GSQFC \notin S_l) = Pr(P = U_j) \cdot Pr(GSQFC \notin S_l) \tag{4.17}$$

$$\begin{aligned} Pr(P = U_j) &= \sum_{j=1, U_i \in S_l}^{b} Pr(U_j \mid M = S_l) \cdot Pr(M = S_l) \\ &= \sum_{j=1, U_i \in S_l}^{b} \frac{1}{K} \cdot \frac{1}{b} = \frac{r}{(b \cdot K)} \end{aligned} \tag{4.18}$$

$$Pr(P = U_j) = \frac{r}{b \cdot K} \tag{4.19}$$

$$Pr(GSQFC \notin S_l) = \frac{(b - r)}{b} \tag{4.20}$$

Equating 4.19 and 4.20

$$Pr(P = U_j, GSQFC \notin S_l) = \frac{r}{b \cdot K} \cdot \frac{(b - r)}{b} \tag{4.21}$$

$$\begin{aligned} &Pr(P = U_j, GSQFC \notin S_l \mid S = U_i, M = S_l) = \\ &Pr(P = U_j \mid S = U_i, M = S_l) \cdot Pr(GSQFC \notin S_l \mid S = U_i, M = S_l) \end{aligned} \tag{4.22}$$

$$Pr(GSQFC \notin S_l \mid S = U_i, M = S_l) = \frac{(b - r)}{b} \tag{4.23}$$

$$Pr(P = U_j \mid S = U_i, M = S_l) = \frac{1}{K} \tag{4.24}$$

Equating (4.23) and (4.24)

$$Pr(P = U_i, GSQFC \notin S_l \mid S = U_i, M = S_l) = \frac{1}{K} \cdot \frac{(b-r)}{b} \tag{4.25}$$

Equation 4.16, 4.21, and 4.26

$$= \frac{r}{(b-1) \cdot K} \tag{4.26}$$

Equation (4.26) shows the probability of linking query with the source when single GSQFC is supposed to forward the queries of all user from multiple to the WSE.

(b) Source and GSQFC belong to the same group

When overlapping group design is used, GSQFC and source occur in the same group, two case may occur, i.$GSQFC = U_j$ ii. $GSQFC \neq U_j$. The probability of such cases is given by.

Case 1

When $GSQFC$ and the query source user occur in the same group such that $GSQFC = U_j$:

$$\begin{aligned} &Pr(S = U_i \mid M = S_l, P = U_j, GSQFC \in S_l) \\ &= \frac{(Pr(S = U_i \mid GQFN \in S_l) Pr(M = S_l, P = U_j \mid S = U_i, GSQFC \in S_l))}{Pr(M = S_l, P = U_j \mid GSQFC \in S_l)} \end{aligned} \tag{4.27}$$

$$\begin{aligned} &Pr(M = S_l, P = U_j \mid S = U_i, GSQFC \in S_l) = \\ &Pr(M = S_l \mid S = U_i, GSQFC \in S_l) \\ &\cdot Pr(P = U_j \mid M = S_l, S = U_i, GSQFC \in S_l) \end{aligned} \tag{4.28}$$

$$Pr(M = S_l \mid S = U_i, GSQFC \in S_l) = \frac{1}{\lambda} \tag{4.29}$$

As U_i and $GSQFC$ are pair in λ groups so

$$Pr(P = U_j \mid M = S_l, S = U_i, GSQFC \in S_l) = \frac{1}{K-1} \tag{4.30}$$

$$Pr(S = U_i \mid GSQFC \in S_l) = \frac{Pr(S = U_i \cap GSQFC \in S_l)}{Pr(GSQFC \in S_l)} \tag{4.31}$$

$$= \frac{Pr(S = U_i \cap GSQFC \in S_l)}{\sum_{i=1}^{b} Pr(S_l) \cdot Pr(GSQFC \mid S_l)} \tag{4.32}$$

$$Pr(S = U_i \cap GSQFC \in S_l) = \frac{\lambda}{k \cdot K} \tag{4.33}$$

$$\sum_{i=1}^{b} Pr(S_l) \cdot Pr(GSQFC \mid S_l) = \sum_{1=1}^{b} \frac{1}{b} \cdot \frac{1}{k} = \frac{r}{b \cdot k} \tag{4.34}$$

Equating 4.33 and 4.34

$$Pr(S = U_i \mid GSQFC \in S_l) = \frac{\lambda}{r} \tag{4.35}$$

$$\begin{aligned} &Pr(M = S_l, P = U_j \mid GSQFC \in S_l) = \\ &Pr(M = S_l \mid P = U_j, GSQFC \in S_l) \cdot Pr(P = U_j \mid GSQFC \in S_l) \end{aligned} \tag{4.36}$$

$$Pr(M = S_l \mid P = U_j, GSQFC \in S_L) = \frac{1}{r} \quad as\ GSQFC = U_j \tag{4.37}$$

$$Pr(P = U_j \mid GSQFC \in S_l) = 1 \quad as\ U_j = GSQFC \tag{4.38}$$

Equating 4.29, 4.30, 4.35 and 4.37 we get

$$Pr(S = U_i \mid M = S_l, P = U_j, GSQFC \in S_l) = \frac{1}{K-1} \tag{4.39}$$

When $GSQFC$, User U_i and belongs to the same group the probability of linking query with the source is given in (4.39).

Case 2

When overlapping group design is used, such that ($GSQFC \in S_l$) and $U_j \neq GSQFC$) then the probability is given by

$$\begin{aligned} &Pr(S = U_i, M = S_l \mid P = U_j, GSQFC \in S_l) = \\ &\frac{Pr(P = U_j, M = S_l \mid S = U_i, M = S_l) \cdot Pr(S = U_i, M = S_l)}{Pr(P = U_j, GSQFC \in S_l)} \end{aligned} \tag{4.40}$$

$$Pr(S = U_i, M = S_l) = Pr(S = U_i) \cdot Pr(M = S_l \mid S = U_i) \tag{4.41}$$

$$\begin{aligned} Pr(S = U_i) &= \sum_{i=1}^{b} Pr(M = S_l) \cdot Pr(U_i \mid M = S_l) \\ &= \sum_{i=1}^{b} \frac{1}{b} \cdot \frac{1}{k} \\ Pr(S = U_i) &= \frac{r}{b \cdot k} \end{aligned} \tag{4.42}$$

$$Pr(M = S_l \mid S = U_i) = \frac{r}{b} \tag{4.43}$$

Equating 4.42 and 4.43

$$Pr(S = U_i, M = S_l) = \frac{r}{b} \cdot \frac{r}{b \cdot k} \tag{4.44}$$

$$Pr(P = U_j,\ GSQFC \in S_l) = Pr(P = U_j) \cdot Pr(GSQFC \in S_l) \tag{4.45}$$

$$\begin{aligned} Pr(S = U_j) &= \sum_{j=1}^{b} Pr(M = S_l) \cdot Pr(U_j \mid M = S_l) \\ &= \sum_{j=1}^{b} \frac{1}{b} \cdot \frac{1}{k} \\ Pr(S = U_j) &= \frac{r}{b \cdot k} \end{aligned} \tag{4.46}$$

$$Pr(GSQFC \in S_l) = \frac{\lambda}{b} \tag{4.47}$$

Equating 4.46 and 4.47

$$Pr(P = U_j, GSQFC \in S_l) = \frac{\lambda}{b} \cdot \frac{r}{b \cdot k} \tag{4.48}$$

$$\begin{aligned} &Pr(P = U_j, GSQFC \in S_l \in S = U_i, M = S_l) = \\ &Pr(P = U_j \mid S = U_i, M = S_l) \cdot Pr(GSQFC \in S_l \mid S = U_i, M = S_l) \end{aligned} \tag{4.49}$$

$$Pr(P = U_j \mid S = U_i, M = S_l) = \frac{1}{K-1} \quad if\ i = j\ and\ j \neq GSQFC \tag{4.50}$$

$$= \frac{1}{K-2} \quad if\ i \neq j\ and\ j \neq GSQFC \tag{4.51}$$

$$\begin{aligned} &Pr(GSQFC \in S_l \mid S = U_i, M = S_l = \\ &\frac{Pr(M = S_l \mid S = U_i, GSQFC \in S_l) \cdot Pr(GSQFC \in S_l \mid S = U_i)}{Pr(M = S_l \mid S = U_i)} \end{aligned} \tag{4.52}$$

$$Pr(M = S_l \mid S = U_i, GSQFC \in S_l) = \frac{1}{\lambda} \tag{4.53}$$

$$Pr(GSQFC \in S_l \mid S = U_i) = \frac{\lambda}{b} \tag{4.54}$$

$$Pr(M = S_l \mid S = U_i) = \frac{r}{b} \tag{4.55}$$

equating 4.53, 4.54 and 4.55

$$Pr(GSQFC \in S_l \mid S = U_i, M = S_l) = \frac{1}{r} \tag{4.56}$$

Equating 4.44, 4.48, 4.50, and 4.56

$$\begin{aligned} &Pr(S = U_i, M = S_l \mid P = U_j, GSQFC \in S_l) \\ &= \frac{1}{\lambda \cdot (K-1)} \quad if\ i = j\ and\ j \neq GSQFC \end{aligned} \tag{4.57}$$

Equating 4.44, 4.48, 4.52, and 4.56

$$= \frac{1}{\lambda \cdot (K-2)} \quad if\ i \neq j\ and\ j \neq GSQFC \tag{4.58}$$

When $GSQFC$ and source belongs to the same group the probability of linking query is given by (4.57) and (4.58) A user can achieve better local privacy (unlinkability) when the source and the GSQFC belong to different groups since it has a low probability of linking the query to the user. However, the best choice is to use a non-overlapping group design as it has the lowest probability of linking the query with the user.

4.2.2 Profile Privacy

The profile privacy validates the level of profile obfuscation by simulating a privacy-preserving protocol. In MG-OSLo, the profile of a user is obfuscated by sending the queries of other group users to the WSE. A privacy metric Profile Exposure Level (PEL) is used to measure the magnitude of profile obfuscation. An experiment performed to measure the level of profile obfuscation, i.e., profile privacy a user achieves by simulating the MG-OSLo. The experiment consists of three steps: i. simulating the MG-OSLo with the dataset mentioned in section 3.4, ii. Building the profile of a user iii. measuring the profile privacy with PEL. The MG-OSLo is simulated for two situations; i) self-query submission allowed, ii) self-query submission not allowed, with the dataset 1 mentioned in Section 3.4.1 and dataset 2 the description of which is given in Section 3.4.2. The experiment is performed by

changing the group size (number of users in a group) and group count (number of groups). The MG-OSLo is simulated with a group size of three users, four users and five users and a group count of three, four, five and six groups. After simulation, the MG-OSLo is next step of the experiment is to build the profile of each user, Section 3.5.2 shows the profile building process, steps like morpho-synthetic and semantic analysis are followed to build the profile of each user in datasets. Computing PEL is the last step of the experiment, PEL measures the difference between the original profile (sending queries directly to WSE without simulating a protocol) and obfuscated profile (after simulating MG-OSL0) of a user using equation 2.5.

4.3 Results and Discussion

This section gives a detail description of profile privacy analysis of MG-OSLo. As mentioned above, the MG-OSLo is simulated for two situations, i.e., i) self-query submission allowed, ii) self-query submission not allowed. Both situations are simulated with dataset 1 and dataset 2. The profile privacy of a user executing MG-OSLo is compared with state-of-the-art three privacy-preserving protocols i.e. co-utile, UUP(e) and OSLo. The comparison are done over privacy metric PEL at degree 1 to degree of ODP hierarchy.

4.3.1 Profile Privacy of MG-OSLo: Self-Query Submission not Allowed Dataset 1

Figure 4.3 (A-D) show the PEL of a user executing MG-OSLo, when a user only forwards other users' queries only, i.e., self-query submission is not allowed. The results show that the average PEL is 55% at first degree, 16% at the second degree, 8.7% at third degree and 8.5% at the fourth degree. The results show that the group size and a number of groups have little effect on PEL. If the peer does not forward his own query, an attacker can exclude him as a query originator thus reducing the anonymity set.

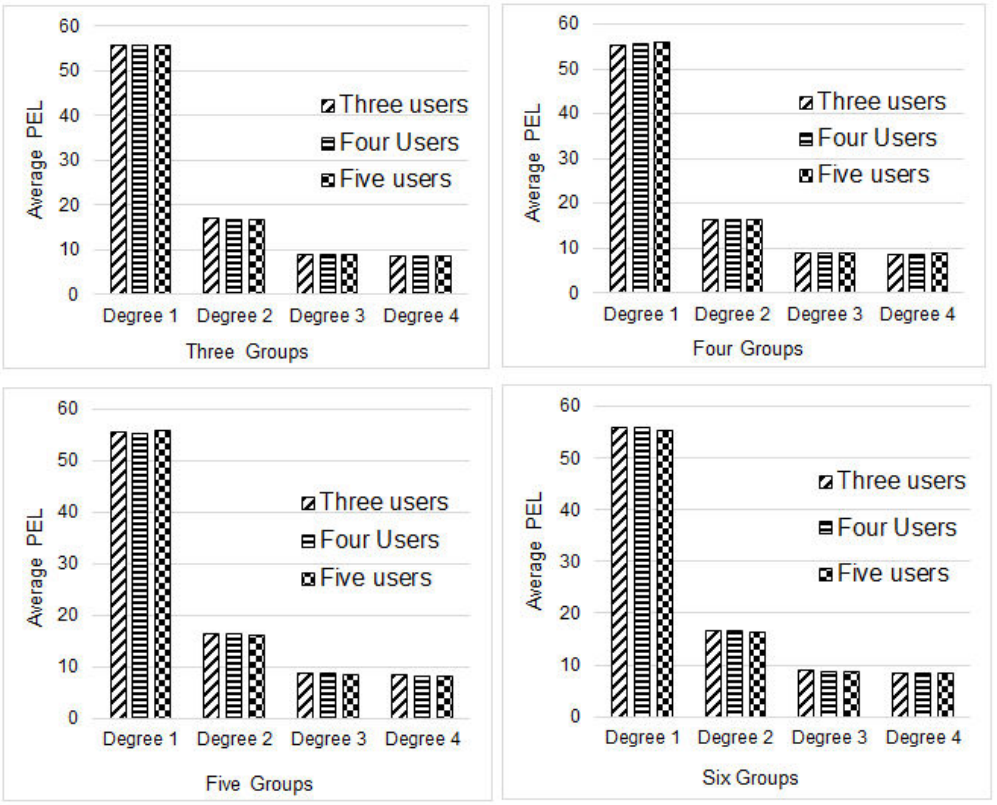

FIGURE 4.3: Average PEL of MG-OSLo: self-query submission not allowed with dataset 1

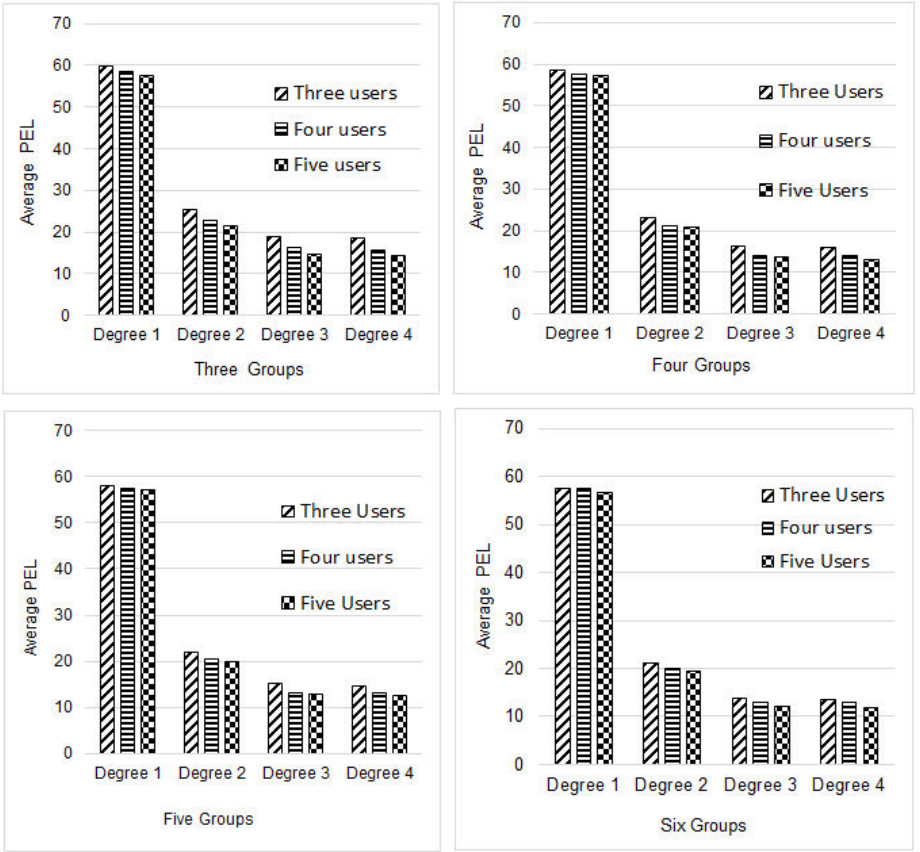

FIGURE 4.4: Average PEL of MG-OSLo: self-query submission allowed Dataset1

4.3.2 Profile Privacy of MG-OSLo: Self-Query Submission Allowed Dataset 1

Figure 4.4 (A-D) show the results when a user selected as GSQFC, can forward his query and queries of the other users. The results show that the PEL of a user is affected by the size of the group and group count. The MG-OSLo successfully hides the profile of a user by more than 40% at the first degree as it represents a more general category. However, at higher degrees, it manages to hide the profile by more than 80%; when the number of peers increases in the group, the corresponding PEL decreases. Increasing the number of groups has a minor effect on PEL as shown in Figure 4.4.

4.3.3 Profile Privacy of MG-OSLo: Self-Query Submission Allowed Dataset 2

Figure 4.5 (A-D) shows the average PEL of a user executing MG-OSLo for a situation in which the self-query submission is allowed. The results show that when the number of users in a group increases, the PEL drops. However, increasing the group count has little effect on PEL. The average PEL MG-OSLo when simulated for three group having three users in each group, is 54.48%; when each group has four users the average PEL is 53.57% and for five users the average PEL is 53.05% at the first degree of the ODP hierarchy. However, at the second degree of the ODP hierarchy, the average PEL is less than 20% for all group sizes and the group count.

4.3.4 Profile Privacy of MG-OSLo: Self-Query Submission not Allowed Dataset 2

Figure 4.6 (A-D) shows the average PEL of a user execution MG-OSLo when simulated with dataset 2 for a situation in which self-query submission is not allowed. The average PEL for the group size of three users, four users and five users at degree 1 of the ODP hierarchy for the group count of three, four, five and

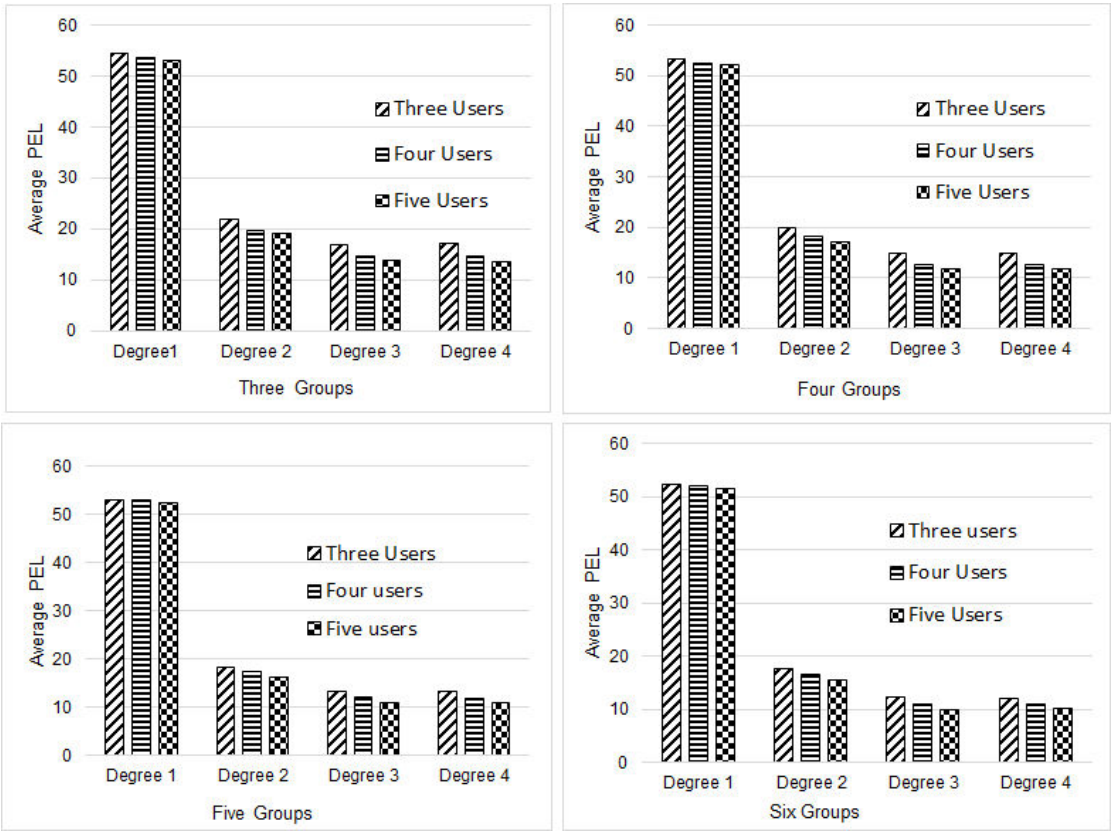

FIGURE 4.5: Profile Privacy of MG-OSLo: Self-query submission allowed Dataset 2

six groups are almost 50%. The group size and count have very little effect on the profile privacy of a user when MG-OSLo is simulated with dataset 2.

4.3.5 MG-OSLo VS OSLo VS Co-utile: Self-Query Submission Allowed Dataset 1

Figure 4.7 and Figure 4.8 show the profile privacy a user achieves relative to the WSE. The values show the average PEL comparison at degree 1 to degree 4 of the ODP hierarchy. The results show that the average PEL of a user executing OSLo for a group size of three users at the first degree of the ODP hierarchy is 67.59%, and 74.58% for the co-utile protocol, whereas the average PEL of MG-OSLo for a group count of three, each group having three users is 59.61%. The MG-OSLo preserved 11.76% and 19.9% better profile privacy as compared to OSLo and co-utile protocols. Similarly, the average PEL further drops by increasing the group count. When the group count increased to four, the MG-OSLo achieved 58.61% average PEL, the value of average PEL further dropped to 58.17% and 57.42%

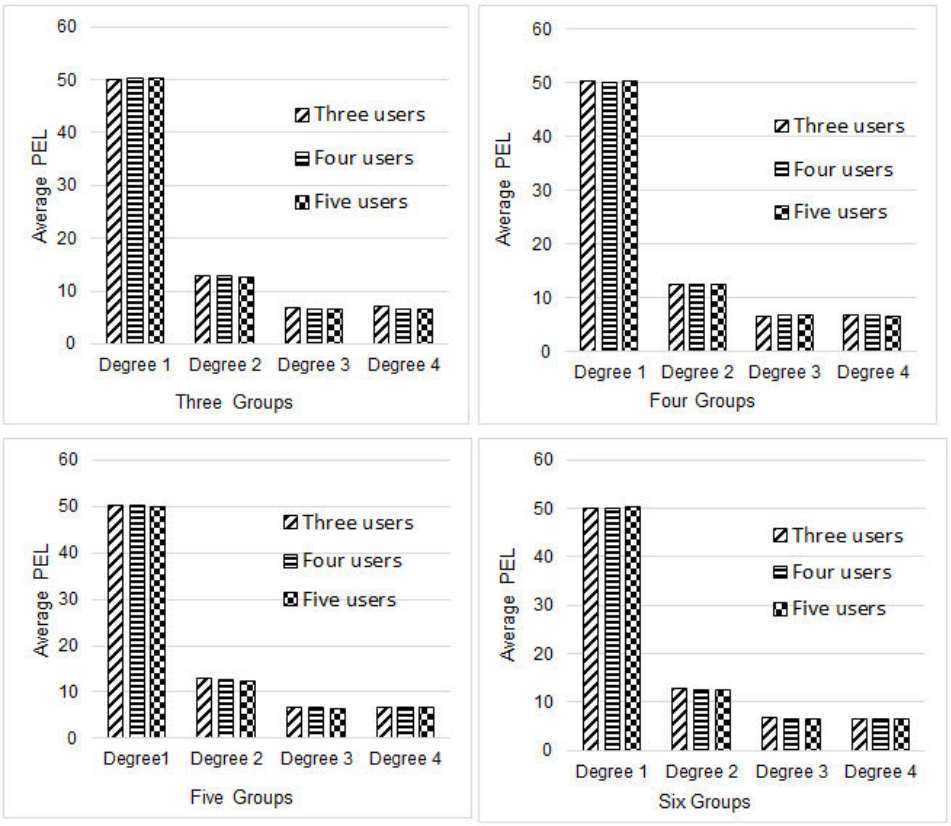

FIGURE 4.6: Average PEL of MG-OSLo: self-query submission not allowed with Dataset 2

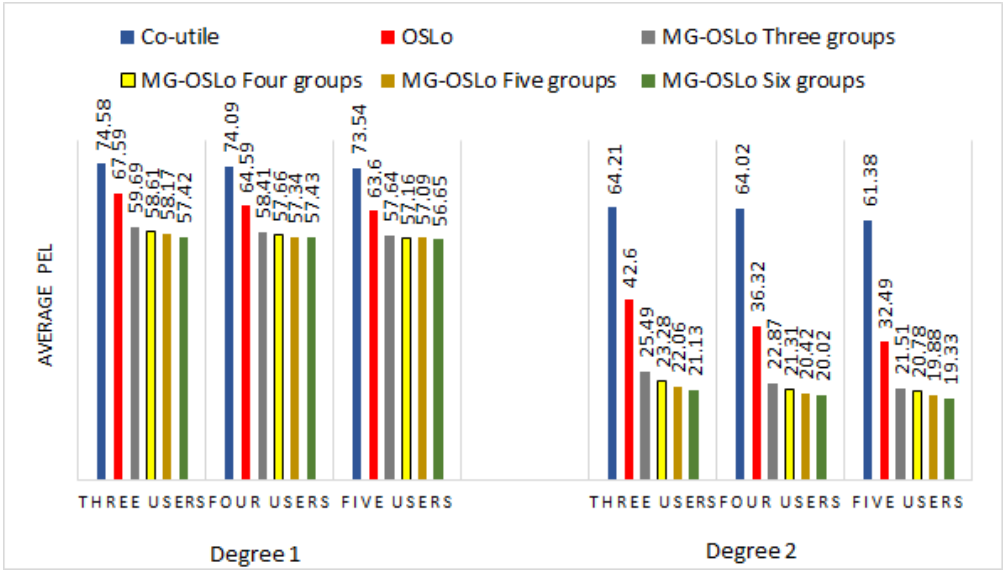

FIGURE 4.7: Average PEL of MG-OSLo VS. OSLo VS. Co-utile at Degree 1 and Degree 2 of ODP hierarchy for Dataset 1

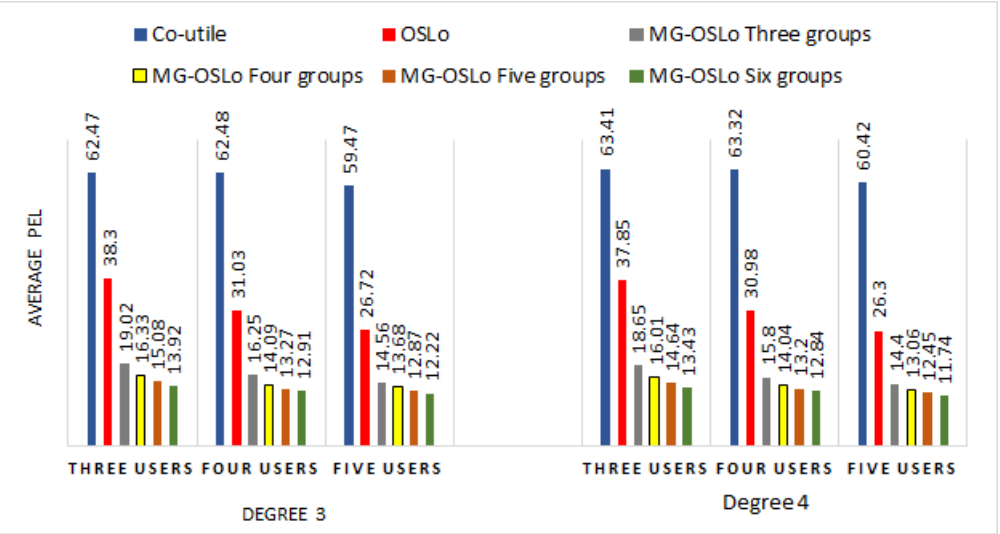

FIGURE 4.8: Average PEL of MG-OSLo VS. OSLo VS. Co-utile at Degree 3 and Degree 4 of ODP hierarchy for Dataset 1

for a group count of five and six, hence, achieving 13.28%, 13.94% and 15.04% better profile privacy as compared to OSLo and 21.4%, 21.9%, and 23.0% better profile privacy as compared to co-utile protocol. When the size of the group is increased to four users, the results depicted that the OSLo achieved 64.59%, and the co-utile protocol acquired 74.09% average PEL, whereas the MG-OSLo with the group count of three procured 58.42%. When the group count increased to four 57.66% average PEL was obtained. The value of average PEL further dropped to 57.34%, and 57.43% for the group count of five and six. In such case when the self-query submission was allowed, the MG-OSLo attained 9.5%, 10.72%, 11.22%, and 11.08% better results as compared to OSLo at degree 1 of the ODP hierarchy and 21.15%, 22.17%, 22.60%, and 22.47% better average PEL as compared to the co-utile protocol. Additionally, the average PEL of a user when five users are grouped, the average PEL for OSLo was 63.60 and for co-utile was 73.54%, whereas, the MG-OSLo with the group count of three 57.66%, 57.16% for the group count of four, 57.08% and 56.65% for the group count of five and six. The comparison shows that MG-OSLo provided better profile privacy at all degrees of the ODP hierarchy for any group size.

The average PEL a user achieves at degree 2 of the ODP hierarchy for the group size of three users is mentioned in Figure 4.7. The user executing OSLo had 46.60% and 64.21% for the co-utile protocol, whereas the user executing the MG-OSLo

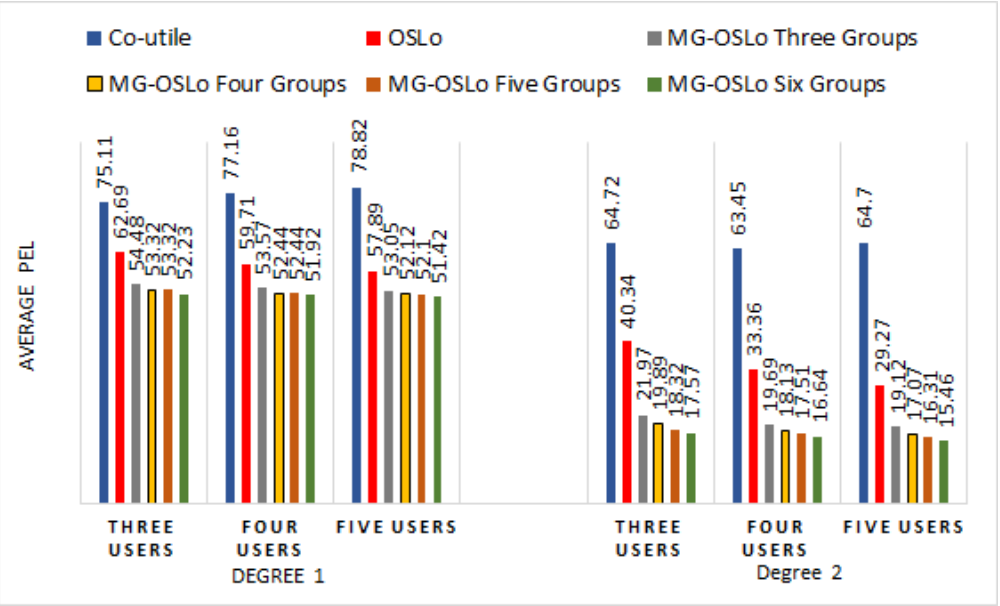

FIGURE 4.9: Average PEL of MG-OSLo VS OSLo VS Co-utile at Degree 1 and Degree 2 of ODP hierarchy for Dataset 2

the average PEL with the group count of three has 25.49%. Similarly, 23.28%, 22.05% and 21.13% for the group count of four, five and six. Likewise, the average PEL of a user executing MG-OSLo had better profile privacy at all degrees of the ODP hierarchy when the self-query submission was allowed.

4.3.6 MG-OSLo VS OSLo VS Co-utile: Self-Query Submission Allowed for Dataset 2

Figure 4.9 and Figure 4.10 show the profile privacy comparison of OSLo, co-utile, and MG-OSLo based on the average PEL when simulated with Dataset 2 for a situation where self-query submission is allowed. The results show that the average PEL of a user, when simulated OSLo and co-utile for the group size of three users was 62.69% and 75.11% at degree 1 of the ODP hierarchy, whereas the average PEL with MG-OSLo for group count three is 54.48%, i.e., 13.09% better than OSLo and 27.45% better than co-utile protocol. The result of the average PEL with MG-OSLo for the group count of four was 53.32%; it achieved 14.94% and 29% better profile privacy as compared to OSLo and co-utile protocol. When four users were grouped, the average PEL with OSLo was 59.72% and 77.16% with co-utile, whereas MG-OSLo is 53.57%, 52.44%, 52.34% and 51.92% for the group

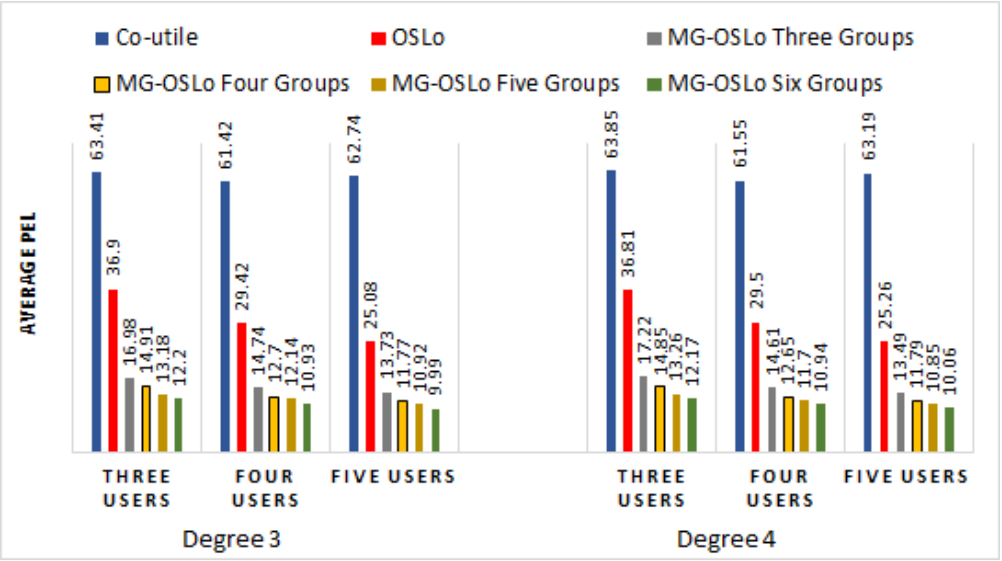

FIGURE 4.10: Average PEL of MG-OSLo VS. OSLo VS. Co-utile at Degree 3 and Degree 4 of ODP hierarchy for Dataset 2

count of three, four, five and six. The average PEL drops when the number of users is increased in the group in both OSLo and MG-OSLo. Furthermore, when the group has five users, the average PEL a user achieved by simulating OSLo is 57.88% and MG-OSLo with the three-group count is 53.05%, 52.11% with a group count of four, and so on. Likewise, with the average PEL at degree 2, degree 3 and degree 4 of the ODP hierarchy, the MG-SOLo has a lower average PEL as compared to OSLo. The MG-OSLo has a better profile privacy as compared to the OSLo for all group counts and group sizes when both protocols are simulated for a situation where self-query submission is allowed.

4.3.7 MG-OSLo vs UUP(e) and OSLo: Self-Query Submission not Allowed Dataset 1

Figure 4.11 and Figure 4.12 shows the profile privacy a user achieve relative to the WSE for a situation when self-query submission is not allowed. The profile privacy a user achieved by simulating MG-OSLo is compared with state-of-the-art protocol UUP(e) and OSLo. All three protocols are simulated for the group size of three users, four users and five users. The MG-OSLo is simulated with the count of three groups, four groups, five groups and six groups. The results show, when three users are grouped together the average PEL of a user executing

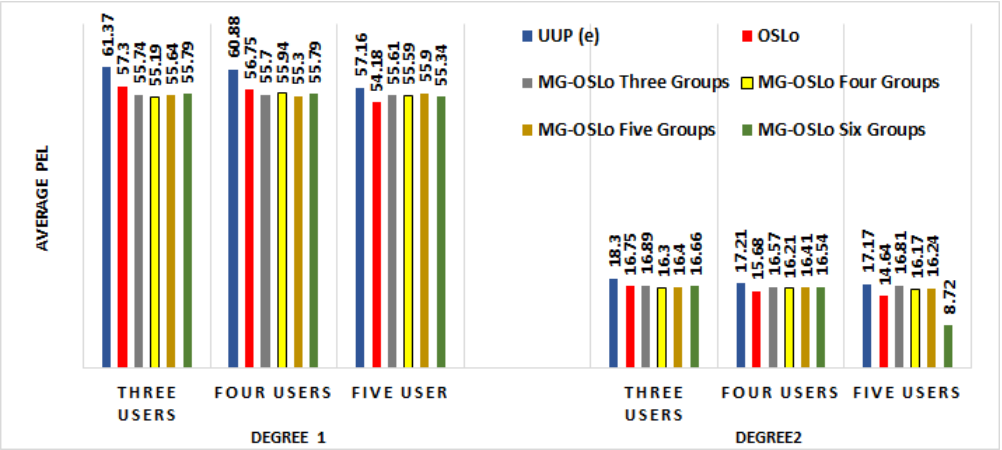

FIGURE 4.11: Average PEL of MG-OSLo VS. UUP(e) and OSLo at Degree 1 and Degree 2 for Dataset 1

UUP(e) is 61.37% and OSLo user has 57.3%. Whereas, the average PEL of a user executing MG-OSLo is 55.74% for the group count of three, 55.19% for the group count of four, 55.65% and 55.79% for the group count of five and six at degree 1 of ODP hierarchy. The MG-OSLo achieves 9.1% better profile privacy as compared to UUP(e) and 3.69% improvement from OSLo for the group count of three users. Similarly, when four users are group together the MG-OSLo attains 8.5% better as compared to UUP(e) and 1.8% as compared to OSLo. The MG-OSLo has improved the profile privacy of a user as compared to the UUP(e) and OSLo for any group size at degree 1 and degree of ODP hierarchy. The reason that MG-OSLo has succeeded in improved profile privacy as compared to UUP(e) and OSLo is that the profile of a user is obfuscated with the queries of a larger number of users present in multiple groups of MG-OSLo.

Although based on the simulation results of allowing the self-query submission it was anticipated that increasing the group count would have compelling impact on profile privacy, however, it is also observed from the simulation results that increasing the group count in MG-OSLo has no significant effect on profile privacy. For example, the difference between average PEL for the group count of three, four, five and six is less than 1% at degree 1 and degree 2 of ODP hierarchy. Therefore, the MG-OSLo with group count of three is recommended when the self-query submission is not allowed. Furthermore, it is also observed the profile privacy at degree 3 and degree 4 (which represents a more specific category) of ODP hierarchy, OSLo preserved slightly better profile privacy (0.71%) than MG-OSLo for the group count and size of three. When the number of users increased to

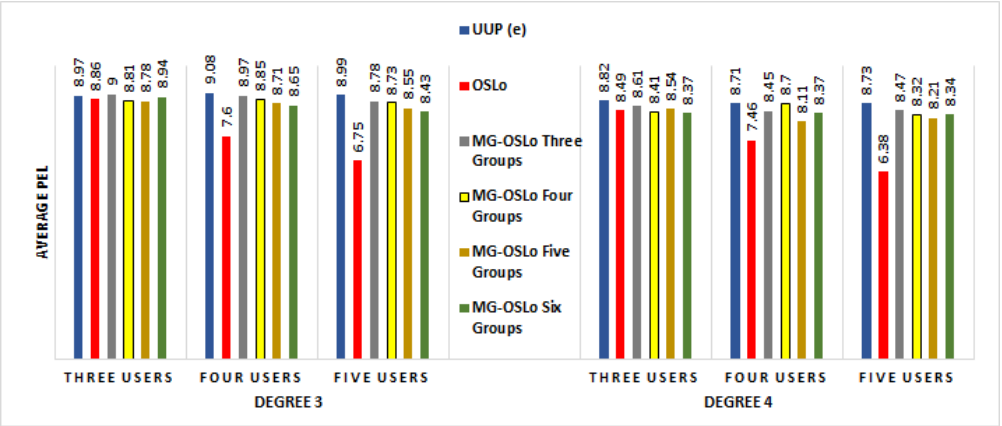

FIGURE 4.12: Average PEL of MG-OSLo VS UUP(e) VS OSLo at Degree 3 and Degree 4 for Dataset 1

four and five the privacy of MG-OSLo decline as compared to OSLo. As discussed above, the reason behind this finding is the profile of the user is obfuscated to its maximum level with group count of three and increasing further the group count and group size has no positive impact on profile privacy.

4.3.8 MG-OSLO VS UUP(e) and OSLo: Self-Query Submission not Allowed at Dataset 2

Figure 4.13 and Figure 4.14 shows the comparison of profile privacy a user achieve by simulating UUP(e), OSLo and MG-OSLo over dataset 2 for a situation when self-query submission is not allowed. The results show when three users are grouped together, the average PEL a user achieve by executing UUP(e) is 51.85%, and OSLo attains 47.69%. The average PEL of a user executing MG-OSLo for group count of three is 47.50%. The MG-OSLo provides 8.3% and 0.40% better profile privacy as compared to UUP(e) and OSLo at degree 1 of ODP hierarchy. When four users are grouped together the average PEL a user reaches by executing UUP(e) and OSLo is 51.15% and 48.56% whereas the MG-OSLo for three group count is 48.29%. The MG-OSLo with three group count provides 5.59% and 0.55% better profile privacy as compared to the UUP(e) and OSLo. It is also noted that when the group count in MG-OSLo is increased the average PEL unexpectedly slightly decreases.

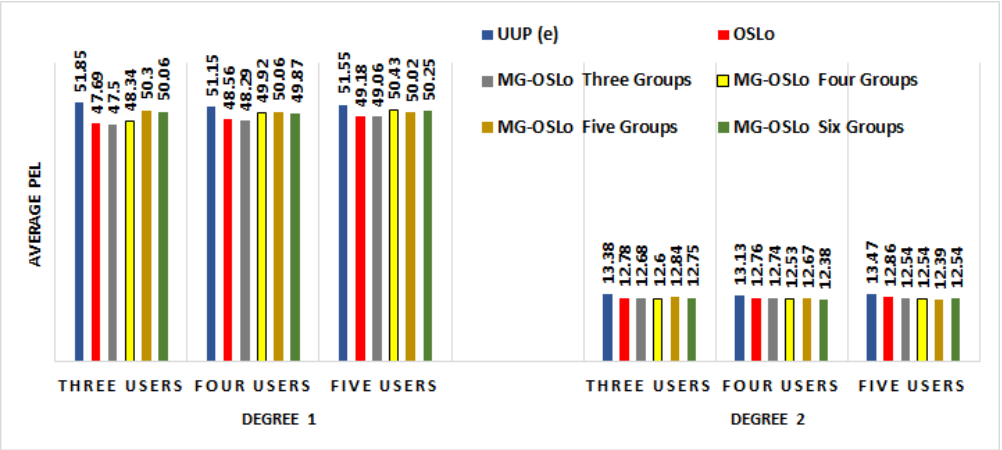

FIGURE 4.13: Average PEL of MG-OSLo VS. UUP(e) VS. OSLo at Degree 1 and Degree 2 for Dataset 2

The results infer that with a group count of three each containing three users achieves the minimum average PEL because the profile of a user is obfuscated to its maximum level in a situation when self-query submission is not allowed. It is observed that increasing the group count or group size any further has no significant impact on average PEL, instead, the average PEL slightly increases in contrast to the situation when the self-query submission was allowed. Therefore, based on the result it is recommended that the group count of three and each group having three users provide the best results in terms of average PEL for a situation when self-query submission is not allowed. However, a user can forward more queries with group count of four and five with a little compromised on average PEL. As delay is a prime concern in any distributed protocol so with higher group count a user can forward many queries, otherwise the whole group creatation process are to be repeated what would cost more delay.

4.3.9 Time Complexity of MG-OSLo

MG-OSLo consists of M number of groups while each group has n users. The connection process happens in $O(n * M)$, as CS has to enqueue "n" number of users in each group and there is a M number of groups. In the group search query forwarding client (GSQFC) selection process, the CS selects a user from each group as GSQFC, this process completes in O(n) time. Similarly, the CS forward a get_GSQFC_info_MSG message to the selected users in O(n) times.

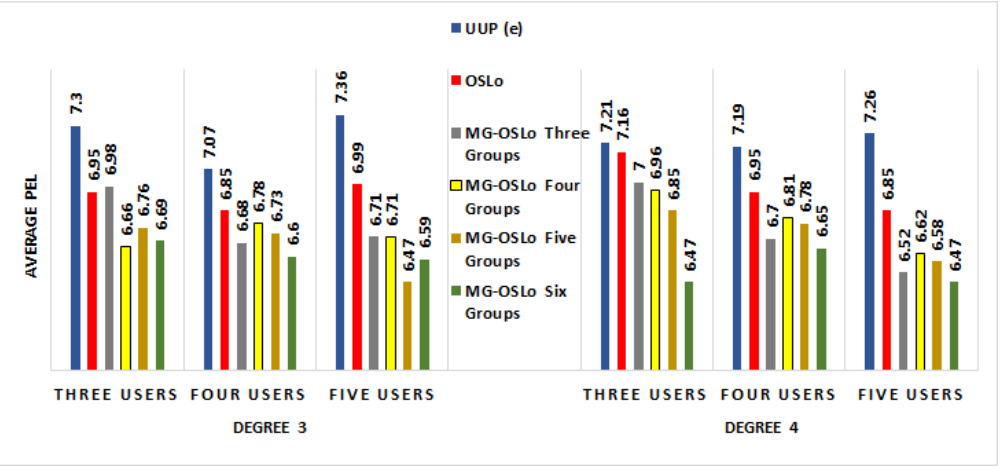

FIGURE 4.14: Average PEL of MG-OSLo VS. UUP(e) VS. OSLo at Degree 3 and Degree 4 for Dataset 2

The CS then broadcasts the group information and the GSQFC to all users in O(n) time. Furthermore, when the CS receives the eQ_Msg or $eAns\_Msg$, it forwards those messages in O(n) times. On the client-side, the query sending process like query content generation, query encryption, query shuffling, and result decryption are sequential steps done in constant time. likewise, when the GSQFC receives an encrypted query message, the packet decryption, query forwarding to WSE & result retrieval, result encryption and forwarding encrypted result to WSE are also done sequentially in constant time. In the worst case, the MG-OSLo protocol executes in $O(n * M)$ time. Where n represents the number of users and each user will forward n queries. However, the experimental results show the MG-OSLo performs best with the group count of three each having three users.

4.4 Conclusion

This chapter highlighted the limitation in the existing multi-group distributed protocols and proposed a novel multi-group distributed privacy-preserving protocol MG-OSLo (Multi Group ObScure Logging). MG-OSLo eliminates the limitation mentioned at the beginning of the chapter and preserves the privacy of a user in a web search. The MG-OSLo preserves the web search privacy of a user using steps like query encryption, shuffling, result encryption and result broadcasting. Previous multi-group protocols only evaluated the privacy of a user relative to the group user; however, the MG-OSLo evaluates the privacy relative to group

user (local privacy) and privacy against the profiling of WSE (profile privacy). The grouping is the primary step of the multi-group distributed protocol, and affects the privacy of a user. To comprehensively evaluate the effect of grouping of local privacy, two grouping approaches, i.e., none-overlapping grouping design and overlapping group design are considered in MG-OSLo. The local privacy is measured by computing the probabilistic advantage an entity has in linking a query with the user. In the proposed protocol (MG-OSLo), the user query is encrypted with the public key of the GSQFC, hiding the query from all other entities. In comparison, in the previous protocols the users associated a memory location and were able to see the query contents as the query was encrypted with the shared key. When the non-overlapping block design is adopted, the probability of linking query by GSQFC with the user is computed as $\frac{1}{(b-1)K}$. When multiple GSQFC and CS makes a coalition to link a query to a user the probability is calculated as $\frac{1}{(b-C)K}$. Similarly, in the overlapping group design where the query originating user and GSQFC belong to a different group, the probability of linking the query is $\frac{r}{(b \cdot K)}$, and when the source user and GSQFC belongs to the same group the probability of linking query with the user is $\frac{1}{\lambda(K-1)}$.

An experiment performed over both datasets (dataset 1 and dataset 2) to evaluate the impact of multi-grouping on profile privacy of a user. The experiment is performed for two scenarios, first, self-query submission is allowed and second self-query submission is not allowed. The results obtained are compared with UUP(e) and OSLo, thereby identifying the impact of multi-grouping on profile privacy. The MG-OSLo is simulated with a group count of three, four and five with a group size of 3 users, 4 users and 5 users in each group for two scenarios discussed above. The results show that MG-OSLo preserves 11.76% better privacy as compared to OSLo for the group count of three, each group having three users in the group. Similarly, when the group count is increased, MG-OSLo preserves better privacy as compared to OSLo at all four degrees of the ODP hierarchy for any group size in both datasets. In the second situation where self-query submission is not allowed, the MG-OSLo has improved the profile privacy 9.1% and 3.69% at degree 1 for the group count of three, while each group has three users for dataset 1. Similarly, the experiment shows when performed with dataset 2, the MG-OSLo preserves 8.3%

and 0.4% better privacy as compared to the UUP(e) and OSLO.
This chapter answers the both parts of research question 1 mentioned in Section 1.8 by varying the group count and group size.

Research Question RQ 1. How to improve the local privacy and profile privacy of a user in a private web search?
RQ 1 (a). What will be the effect of allowing self-query submission and not allowing self-query
RQ 1 (b) How does the **size** of the group and **group count** affects the privacy and performance of a user"?

The profile privacy of MG-OSLo is evaluated for allowing self-query submission and not allowing the self-query submission. The impact is stated in the results and the reason for variation in the results are discussed in this chapter. Moreover, The impact of group count and group size on local privacy and profile privacy is also elaborated in this chapter. In this chapter, the research question RQ 1 with it subparts RQ 1(a) and RQ 1(b) are addressed with empirical results.

Chapter 5

Profile aware ObScure Logging (PaOSLo)

The distributed privacy-preserving protocols work based on the cooperation of users. These protocols create a group of 'n' users who wish to query the WSE without using their privacy. Each user in the group forwards a query of another user to the WSE to receive the query result. Grouping of users is considered a primary step in the distributed privacy-preserving protocols. In the existing approaches, the Core Server (CS) listens to the connection request from the user, upon receiving 'n' number of connection requests, the CS creates a group of users on first comes first serve basis. The major shortcoming in the first come first serve approach, also called random grouping, is that a user maybe groups with those users who have similar interests, as there is no prior information about the users' preferences [8]. Consider a situation where a user Mr. X has a medical related query. He is in a group with other users having a similar type of interest; in such a case, although, Mr. X forwards a query of another user, his profile will not be significantly obfuscated. The profile maintained by the WSE for Mr. X will contain the same type of categories. Profile obfuscation is the fundamental objective of web search privacy; however, with existing randomized grouping a user can possibly be grouped with users having similar interests. Such grouping will not expressively obfuscate the profile of the users.

This chapter proposes a novel protocol PaOSLo (Profile aware ObScure Logging)

to significantly obfuscate the profile of a user by grouping users based on their profile dissimilarity as compared to the random grouping approach. The PaOSLo executes in two steps: (i) cluster the user according to their profile similarity. (ii) the CS creates a group of users from each distinct cluster instead of random grouping. The user when selected as SQFC, only forwards the queries of other group users but not of his own query. This approach obfuscates the profile of a user with the queries of a user who has different interests.
Following are the objective of PaOSLo.

1. Profile aware grouping to considerably obfuscate the profile of a user.
   (a) The cosine similarity measures the similarity between the users' profile.
   (b) The K-Mean algorithm group the users into three cluster, four cluster, and five clusters
   (c) Create a group of users by selecting each user from a different cluster.
2. An experiment is performed to compare the level of profile obfuscation using profile aware grouping and randomize grouping over the queries of users described in dataset 2.

5.1 PaOSLo Description

The Profile aware ObScure Logging (PaOSLo) obfuscates the profile of a user with the queries of a user having dissimilar interests. The PaOSLo has the same entities as OSLo; the description of entities and execution process of each entity are discussed in section 3.2 and 3.3. However, before the execution of protocol, the primary aim of PaOSLo is to cluster the users who have a almost similar interests or almost similar profile. The cosine similarity metric is adopted measures the similarity between the users' profile. After calculating the similarity between the users' profile, the next step is to cluster the user based on their profile similarity. The clustering is performed with the K-mean clustering algorithm using the Weka tool.

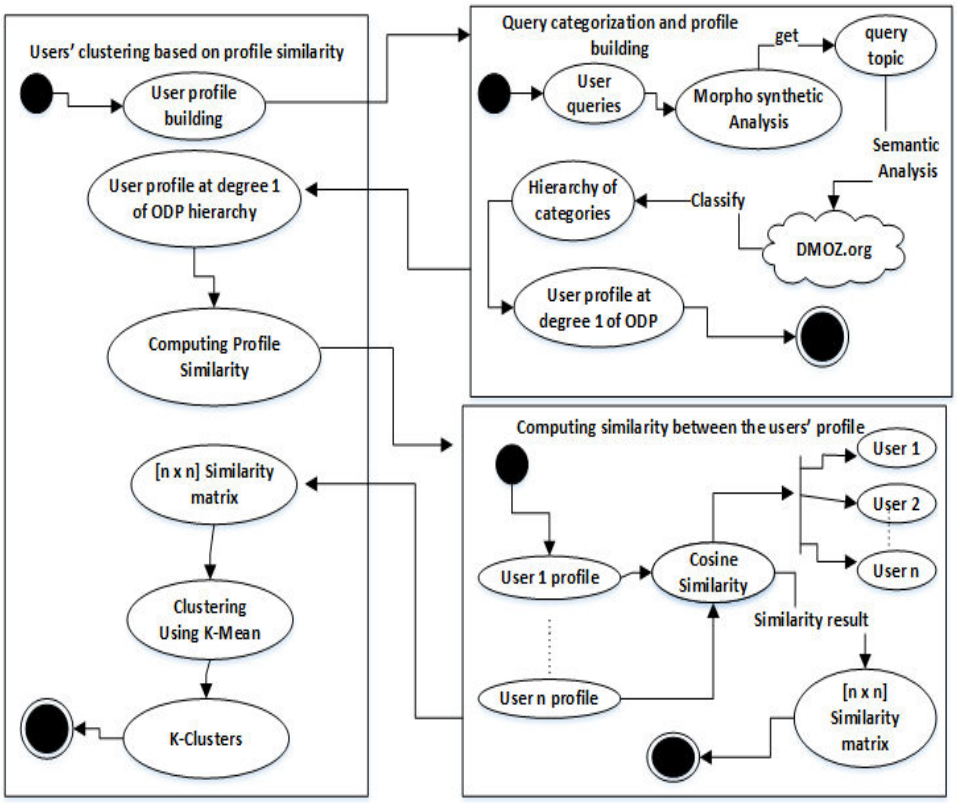

FIGURE 5.1: PaOSLO: Activity diagram of User profile construction and clustering

5.1.1 User Profile Construction

The profile is built from the queries that user send to the WSE. In this experiment, AOL query log described in section 3.4. The AOL query log consists of five attributes:, i.e.,AnonID, Query, Query Time, ItemRank and ClickURL [27, 49]. However, in the experiment of simulating PaOSLo only two attributes, are used, i.e., AnonID and Query. In this work, PaOSLo is simulated with a subset of 1000 queries submitted by users in a three month period of the AOL query log. Section 3.4.2 gives the description of the dataset used in the experimentation. The WSE builds the profile of a user from the queries it receives. Section 3.5.2 explains the profile building process; two fundamental steps are required to build the profile of a user. Figure 5.1 shows the activity diagram of users' profile construction and clustering. The initial step is the to extract the main topic of the query; it involves the morpho-syntactic analysis of the query. The second step performs the semantic analysis of the terms acquired in the previous steps. The terms are sent to DMOZ[1] to classify it into a hierarchy of categories [82]. DMOZ is an

[1]demoz.org, (accessed 6 February 2017)

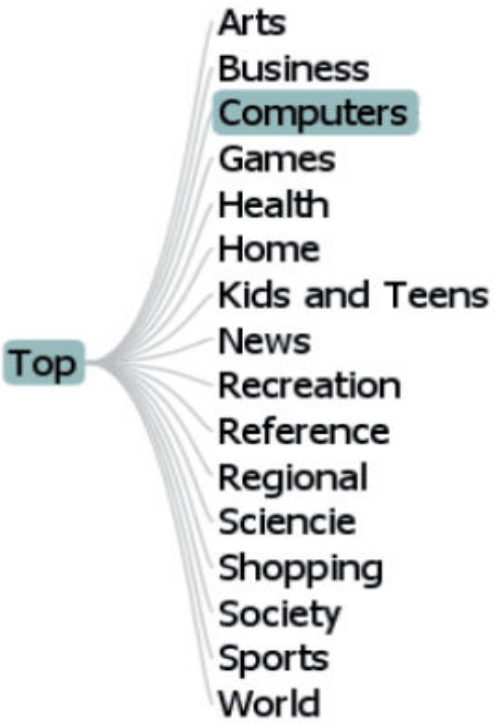

FIGURE 5.2: Categories at first degree of the ODP hierarchy [83]

open-content directory of World Wide Web links, the site and community who maintained DMOZ are also known as the Open Directory Project (ODP). The profile of the users are built from the queries mentioned in Section 3.4.2. The queries of the users are categorized into degrees 1-4 of the ODP hierarchy. Table 5.1 below shows an example of categorization of queries into a hierarchy of degrees by ODP. A query of a user is classified into one of the sixteen categories at the first degree of the ODP hierarchy; Figure 5.2 shows the top level 16 categories at first degree of the ODP hierarchy. The query "valley national bank, photography studio, and mac.com" are categorized as "business, arts, and computer" at first degree of the ODP hierarchy, "financial services, photography, and software" at the second degree of the ODP hierarchy.

5.1.2 Measuring Similarity between the User Profiles

The queries of users selected from the AOL query log mentioned in the dataset 2 are categorized into hierarchy of degrees by ODP. Table 5.2 shows a query categorization of an AOL user denoted with pseudo ID "3978802". The similarity

TABLE 5.1: Query classification of queries by ODP into a hierarchy of categories. [82]

Query	ODP classification at different degrees
Valley National Banker	Business: Financial Services: Banking Services: Credit Unions: Regional: United States:
Photography Studios	Arts: Photography: Techniques and Styles: Documentary: Photographers
mac.com	Computers: Software: Operating Systems: MacOS: Internet

between the users' profiles is computed over the degree 1 of the ODP hierarchy. In this chapter, the following are the two steps taken to compute the similarity between the users' profiles. First, extract the terms at degree 1 from the categorized profile of a user. In the case of user "3978802", Table 5.3 shows the terms and its count at degree 1 of the ODP hierarchy. Similarly, Table 5.4 shows the terms and count of the AOL user pseudo ID "2806175" at degree 1 of the ODP hierarchy. After extracting the terms at degree 1 for all users, the second step is to compute the similarity among the users' profile.

5.1.3 Cosine Similarity

The Cosine similarity measures the similarity between two vectors; it is a function of the angle between their vectors in the term vector space [87]. Equation 5.1 computes the similarity between vector A and B; it gives a value between 0 and 1, where 1 represents exactly the same profiles and 0 presents completely different. Applying equation 5.1, the similarity computed between the profiles of user pseudo ID "3978802" and "2806175" is 0.335. The above-mentioned two steps are followed to compute the similarity between the users' profiles at degree 1 of the ODP hierarchy for the users mentioned in dataset 2. Table 5.5 shows a sample of similarity matrix between six users' profiles; the same way 1000 x 1000 profile similarity matrix is computed for inputting data into Weka tool for clustering.

$$\cos(\mathbf{A},\mathbf{B}) = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n}\mathbf{A}_i\mathbf{B}_i}{\sqrt{\sum_{i=1}^{n}(\mathbf{A}_i)^2}\sqrt{\sum_{i=1}^{n}(\mathbf{B}_i)^2}} \tag{5.1}$$

TABLE 5.2: Query categorization by ODP of a sample user "3978802" of AOL query log

Query	**Degree 1**	**Degree 2**	**Degree 3**	**Degree 4**
map quest	Reference	Maps	Libraries	
la quinta inn	World	Nederlands	Kunst	Schilderkunst
Quinta inn	World	Nederlands	Kunst	Schilderkunst
grove city ohio map	Reference	Maps	Libraries	
yahoo.com games	Society	Religion and Spirituality	Christianity	Denominations
unclaimed funds	Business	Investing	Funds	Mutual Funds
woodworking hobbies	Kids and Teens	Sports and Hobbies	Summer Camps	North America
birdhouse kits	Computers	Speech Technology	Tool Kits	
kazoo toys.com	Arts	Music	Instruments	Winds
tim allen birdhouse	Arts	People	A	Allen 2C Tim
Menards	Recreation	Outdoors	Hunting	Guides and Outfitters
brain cancer gold	Health	Conditions and Diseases	Cancer	Brain and CNS
brain cancer vinyl chloride	Health	Conditions and Diseases	Cancer	Brain and CNS
hc.sc.gc.ca	Regional	North America	Canada	Governmen

TABLE 5.3: Terms extracted for degree 1 of user "3978802"

Terms and Count	Arts	Business	Computers	Games	Health	Home	Kids_and _Teens	News
	2	2	2	0	7	0	1	0
	Recreation	Reference	Regional	Science	Shopping	Society	Sports	World
	1	1	1	0	0	1	0	3

5.1.4 Profile Clustering

Clustering is the process of grouping objects based on their similarity, such that the object in the same cluster is highly similar, whereas it is dissimilar from the objects of other clusters. Profile clustering is the second step of PaOSLo, the similarity between the users' based on degree 1 of the ODP hierarchy calculated

TABLE 5.4: Terms extracted for degree 1 of user "280617"

Terms and Count	Arts	Business	Computers	Games	Health	Home	Kids_and _Teens	News
	0	6	5	1	0	3	0	2
	Recreation	Reference	Regional	Science	Shopping	Society	Sports	World
	0	1	4	0	0	1	0	0

TABLE 5.5: Similarity between sample seven users' profile at degree 1 of ODP hierarchy

User to User cosine similarity	1000335	1002092	100218	1002425	1002791	1003275	1004684
1000335	1	0.675079	0.683216	0.659781	0.852979	0.428317	0.73927
1002092	0.675079	1	0.69809	0.657131	0.503999	0.532601	0.741844
100218	0.683216	0.69809	1	0.802557	0.694908	0.348226	0.522755
1002425	0.659781	0.657131	0.802557	1	0.523591	0.316964	0.59288
1002791	0.852979	0.503999	0.694908	0.523591	1	0.271875	0.677567
1003275	0.428317	0.532601	0.348226	0.316964	0.271875	1	0.46275
1004684	0.73927	0.741844	0.522755	0.59288	0.677567	0.46275	1

TABLE 5.6: Number of users in each cluster after K-Mean clustering

Cluster count	Cluster 1	Cluster 2	Cluster 3	Cluster 4	cluster 5
Three cluster	390	306	304	-	-
Four Cluster	266	188	311	235	-
Five Cluster	241	122	222	236	179

in the previous step works as an input for profile clustering. Weka tool cluster the users' based on profile similarity [88]. To perform the process of clustering, the profile similarity matrix computed in the previous step is transformed into Attribute-Relation File Format (ARFF). An ARFF is the extension of Comma Separated Value (CSV), this file is an ASCII text file that defines a list of instances sharing a set of attributes [89]. ARFF files consist of Header information and Data information. The first Section (Header) holds the name of the relation, the list of attributes and the type of attribute (numeric, nominal, string, and date & time). The data Section defines the data declaration and actual instances lines. Figure 5.3 shows a sample of ARFF file, such type of file is created and loaded into Weka. The profile of the users' is grouped into three cluster, four clusters, and five clusters using a simple K-Mean algorithm. Euclidean distance is used to calculate the distance between the objects of the clusters [90]. Based on the minimum value of Euclidean distance each object is assigned to one of the clusters. Table 5.5 shows the number of users assigned to each cluster; the cluster is calculated over the term count of degree 1 categories of the ODP hierarchy.

Figure 5.4 shows the term count in each category of degree 1 in the ODP hierarchy.

```
@relation profile_data_clustered

@attribute Instance_number numeric
@attribute 1000335 numeric
@attribute 1002092 numeric
@attribute 100218 numeric
@attribute 1002425 numeric
@attribute 1002791 numeric
.
.
.
@attribute Cluster {cluster0     cluster1          cluster2}

@data
0,1,0.675079,0.683216,0.659781,0.852979,0.428317,0.73927
1,0.675079,1,0.69809,0.657131,0.503999,0.532601,0.741844
2,0.683216,0.69809,1,0.802557,0.694908,0.348226,0.522755
3,0.659781,0.657131,0.802557,1,0.523591,0.316964,0.59288
4,0.852979,0.503999,0.694908,0.523591,1,0.271875,0.677567
```

FIGURE 5.3: Sample ARFF file

The x-axis of the graph shows the top level 16 categories of ODP, whereas the y-axis shows the count of the terms in each cluster. Cluster 1 comprises those users who have sent queries mostly from "Regional" category. Cluster 2 has users who have sent most queries from "Business" category and cluster 3 has users who have mostly sent queries from the "Arts" categories. Similarly, when users grouped in four clusters, cluster 1 contains mostly those users who have forwarded queries from the "Arts" category. Cluster 2 contains users who have interests in "Business" categories, cluster 3 contains users of "Regional" category, and cluster 4 from "Computer" category.

Figure 5.5 shows when users are grouped into four clusters based on the term count of degree 1 of the ODP hierarchy. Cluster 1 contains most users that have sent queries from the "Art" category and users of cluster 2 have interests in the "Business" category. Similarly, cluster 3 contains predominantly queries from the "Regional" category and cluster 4 contains queries from the "Computer" category. Figure 5.6 shows the term count of each category of ODP at degree 1 for a situation when users are grouped into five clusters. Cluster 1 contains those users who have sent maximum queries from the "Arts" category; "Computer" category dominates

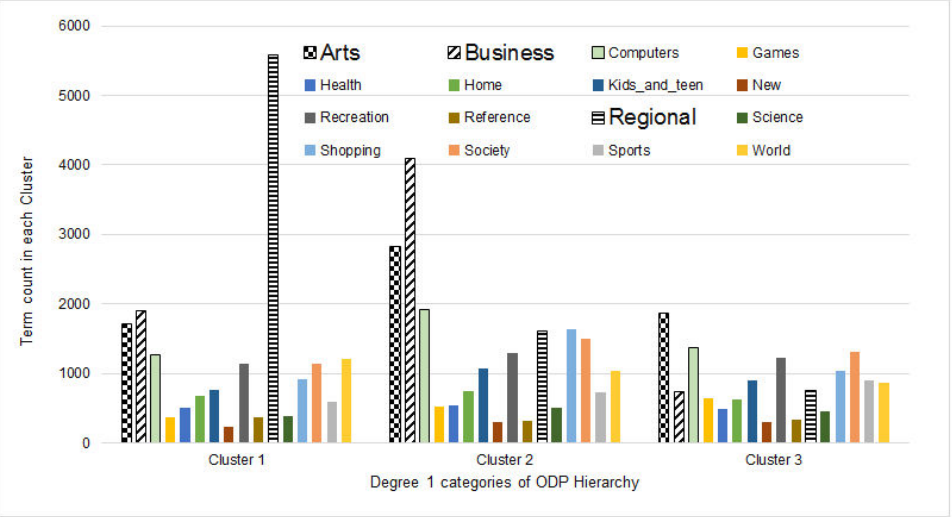

FIGURE 5.4: Term count in three cluster

cluster 2, cluster 3 with "Regional", cluster 4 with"Recreation" while cluster 5 is dominates with "Business" categories.

Based on profile similarity, once the users are clustered, the next step of PaOSLo is to create a group with CS. The PaOSLo has the same entities as OSLo discussed in Section 3.2. The group creation process, SQFC selection, query sending process, query shuffling, query forwarding to the WSE, and result-broadcasting process and algorithm for each process are discussed in Section 3.3. The major difference between the profile aware grouping (PaOSLo) and random grouping (OSLo) is the prior clustering of users based on profile similarity. As PaOSLo is simulated with the group size of 3 users, 4 users and 5 users so datasets are organized into three clusters, four clusters and five clusters to evaluated the impact of clustering.

5.2 PaOSLo Execution Process

The following are the necessary steps required in the execution of PaOSLo.

1. The primary step of PaOSLo is to compute the similarity between the users' profile at degree 1 of the ODP hierarchy using cosine similarity measure.

2. The second step is to cluster the users' having similar interest, K-Mean algorithm clusters the users into three, four, and five clusters.

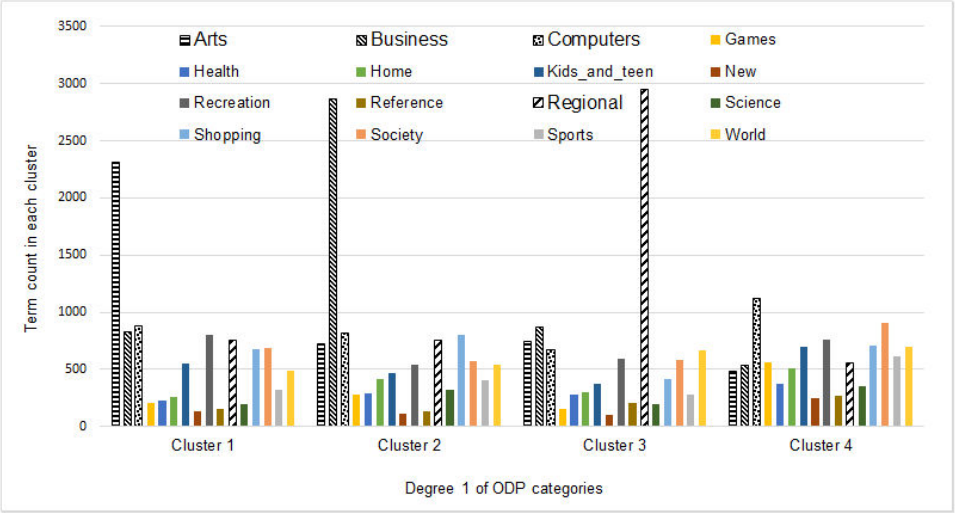

FIGURE 5.5: Term count in four cluster

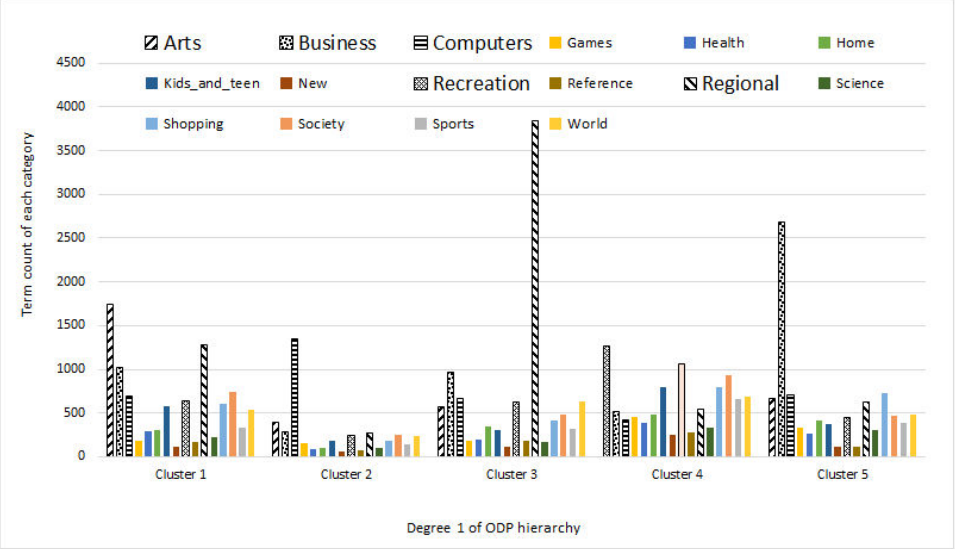

FIGURE 5.6: Term count in five cluster

3. To obfuscate the profile of a user with the queries of dissimilar interest a group of 'n' users is created by selecting one user from each distinct cluster. The PaOSLo is simulated with the group size of three users, four users, and five users. Figure 5.7 shows the activity diagram of group creation and SQFC selection process.

4. The SQFC selection, the query sending process, i.e., encryption, shuffling, decryption, forwarding query to WSE, query result encryption, query result broadcasting, and query result decryption are performed in the same fashion as discussed in Section 3.3. Figure 5.8 shows the activity diagram of query sending and result broadcasting process.

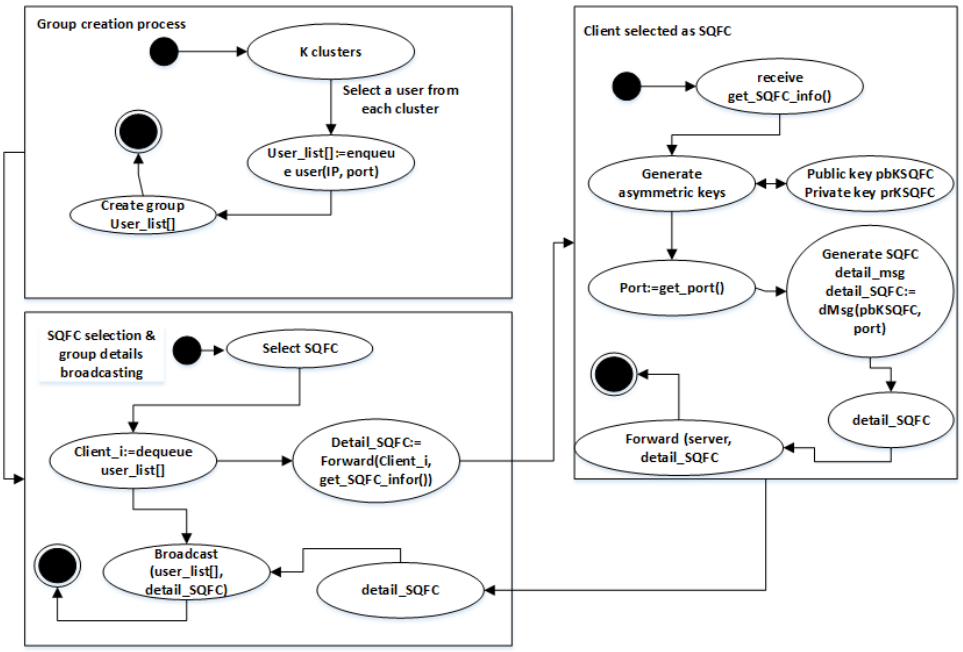

FIGURE 5.7: PaOSLo: Acitivity diagram of Group creation and SQFC selection process

It is important to mention that PaOSLo creates a group of 'n' users by selecting a user from each distinct cluster; after simulation the PaOSLo came across with the situation when one cluster gets empty while other still have users with queries. This is due to users are not evenly distributed in the cluster; rather some cluster has a higher number of users as compared to others. For example, when the dataset is divided into five clusters, cluster 1 has 122 users; cluster 2 has 179 users, and so on. When the PaOSLo is simulated for the group of five users, those five users are selected from five clusters, i.e., one user from each distinct cluster. The cluster with the smaller number of queries/users is exhausted while the other four clusters still contain users with queries. In such a case, the PaOSLo is simulated again, but this time with a group of four users, each user from four different clusters. Similarly, after some more executions by PaOSLo, another cluster is exhausted, and PaOSLo is then simulated for the group size of three users. At one point, the PaOSLo is left with only two clusters with users having only very few queries left. In such a situation, the queries that are not forwarded to the WSE are excluded from the original file of the user. For example, a user had 40 original queries and he has forwarded only 35 queries while 5 queries are yet to be sent and the clusters are exhausted. Those five queries will be excluded from the original file when computing the profile privacy.

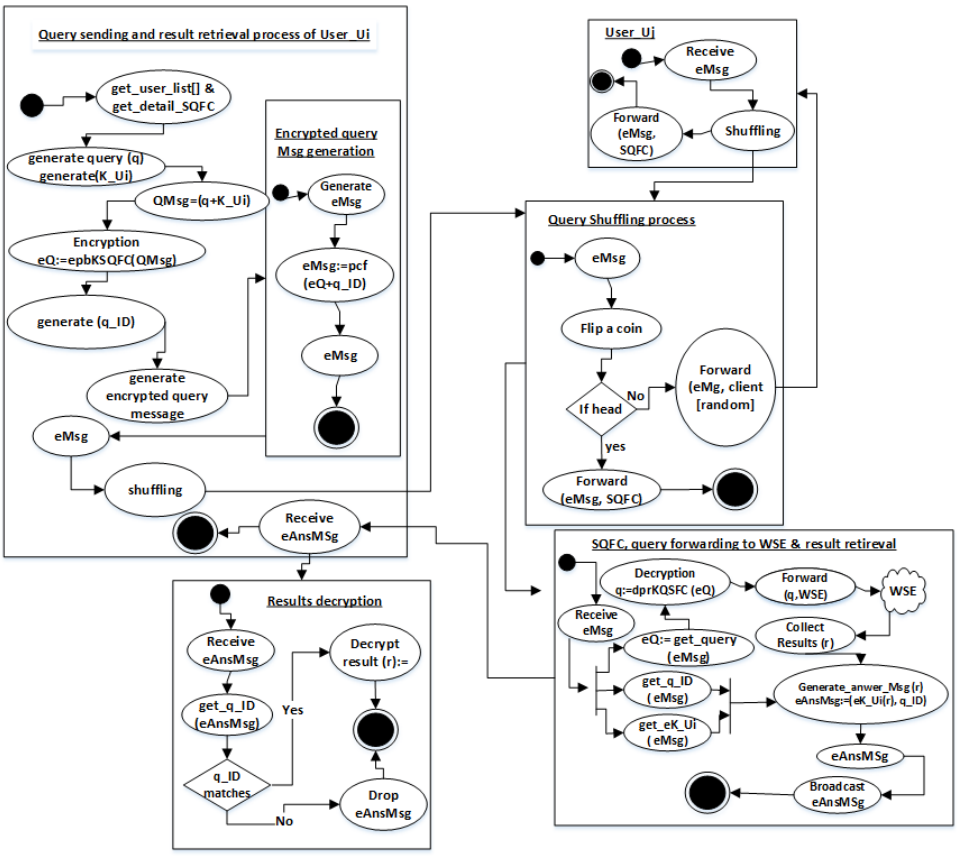

FIGURE 5.8: PaOSLo: Activity diagram of query sending and result broadcasting process

5.3 Privacy of PaOSLo

The privacy objective of a user executing a distributed privacy-preserving protocol is to achieve the following objectives, i.e., the query content and result to query remains hidden from the group entities. Query that cannot be linked with the user will hamper the capability of WSE to build actual profile of the user. As evaluated in previous chapters, the privacy of a user executing PaOSLo will be executed in two dimensions, i.e., the local privacy and profile privacy.

5.3.1 Local Privacy of PaOSLo

A user preserves the local privacy relative to the peer entities involved in forwarding a query to WSE. The local privacy is enforced through query encryption (to hide the contents of query and result), query shuffling (to ensure unlinkability) and result in broadcasting. Section 3.5.2 explains the query encryption process

(detailed in 1); PaOSLo adopts the same approach to encrypt the query content. Similarly, to achieve unlinkability a query is shuffled among the group users, query-shuffling procedure are discussed in (2) of 3.5.2. As the query is encrypted with the public key of SQFC, no user in the group can see the query contents. However, if the SQFC is curious and wants to link a query to the user, the question is what advantages does a SQFC have? What if the SQFC builds a coalition with group users?

Let two random variables S and P, where S denotes the source of the query, and P represents the proxy (a peer user in the group), pass the query to the $SQFC$. Suppose there are 'n' users in the group. If the $SQFC$ wants to find the originating user of a leery query, the probability of linking the query to the user "U_i" is given below.

$$Pr[S = U_i \mid P = U_j] = \frac{Pr[P = U_j \mid S = U_i] \cdot Pr[S = U_i]}{Pr[P = U_j]} \tag{5.2}$$

$$Pr[P = U_j \mid S = U_i] = Pr[P = U_j] \tag{5.3}$$

$$Pr[S = U_i] = \frac{1}{n-1} \tag{5.4}$$

Where, n represents the number of users in a group and $i, j \in (1...n)$, as $SQFC$ is not the query source so $SQFC$ excludes himself.

$$Pr(S = U_i \mid P = U_j) = \frac{1}{n-1} \tag{5.5}$$

Equation (5.5) shows the probability of linking query with the user by $SQFC$ depends on a number of users in a group, all users in the group are equally probable. However, If $SQFC$ and C users collaborate to identify the query originat then the probability of linking query is given in (5.6)

$$Pr(S = U_i \mid P = U_j) = \frac{1}{n-C} \tag{5.6}$$

Equation 5.6 shows, if $SQFC$ makes a coalition with C user the probability of linking query with the initiator $\frac{1}{n-C}$, which means all compromised C users will be excluded from the list. If C is equal to n, it means all users are compromised and there is no one to link the query with, if $n - C$ equal to one means all users are compromised except the originator, $n - C$ shall be greater than one.

The CS and peer users cannot read the content of the query and query results as they are encrypted, however, if any of the curious entity makes a coalition with $SQFC$, then equation (5.6) shows the chances of associating the query with the originator. As the CS is not involved in the query shuffling process, if SQFC does not collaborate with CS or group user, none of the curious entity would see the query or query result and the probability of relating the query with the originator is $\frac{1}{n}$, i.e., all users are equally probable. However, if CS makes a coalition with $SQFC$, the chances of linking the query with the originator are given in Equation (5.5). The group users do not see the query (q) or result returned (r), however, if the compromised peers make a coalition with SQFC then the probability of linking query with the originator is given in Equation (5.6)

5.3.2 Profile Privacy of PaOSLo

Profile privacy measures the privacy of a user relative to the WSE, it computes the difference between the original profile and obfuscated profile of the user. A privacy metric PEL measures the level of the profile exposure from the observance of obfuscated profile. PEL measures the difference between original profile (built from the queries what user send directly to the WSE) and obfuscated profile (built from the queries what user send after the execution of PaOSLo). PEL uses entropy and mutual information according to equation 2.5 to compute he difference between the original profile and obfuscated profile. Table 5.1 shows the average PEL a user achieves by executing PaOSLo. The Average PEL at degree 1 of the ODP hierarchy for the group size of three users is 46.64%, at degree 2 the average PEL drops to 11.14%, the average PEL further drops to 5.5% at degree 3, and so on. Similarly, for the group size of four users, the average PEL is 46.32% at degree 1, 10.6% at degree2, 5.44% at degree 3 and 5.75 at degree 4 of the ODP hierarchy. Likewise, the average PEL for the group size of five users at degree 1 is 46.29%, 10.58%, 5.83% and 5.92% at degree 2, degree 3 and degree 4 of the ODP hierarchy. The results show that the average PEL of a user decreased when the group size is increase. Similar pattern is observed at all degree of ODP hierarchy. This is because the profile of a user is obfuscated with the multiple users having dissimilar interests.

TABLE 5.7: Average PEL comparison of UUP(e), OSLo and PaOSLo

Number of User	Protocol	Degree 1	Degree 2	Degree 3	Degree 4
3 Users	UUP(e)	51.86	13.39	7.3	7.22
	OSLo	47.7	12.79	6.95	7.17
	PaOSLo	46.65	11.14	5.5	6.01
4 Users	UUP(e)	51.16	13.14	7.07	7.2
	OSLo	48.56	12.76	6.85	6.95
	PaOSLo	46.32	10.6	5.44	5.75
5 Users	UUP(e)	51.55	13.47	7.37	7.26
	OSLo	49.18	12.86	6.99	6.85
	PaOSLo	46.29	10.58	5.39	5.92

5.3.3 Profile Privacy Comparison of UUP(e), OSLo and PaOSLo

The profile privacy achieved by a user executing PaOSLo is compared with the state-of-art privacy-preserving protocol UUP(e) and OSLo. This is basically a comparison between random grouping of users and profile aware grouping of users. Table 5.7 shows the result comparison of average PEL of UUP(e), OSLo and PaOSLo. All three protocols employ single dynamic groups and they are simulated for a situation where self-query submission is not allowed, simulated over the same dataset described in section 3.4.2. The results show that PaOSLo has less average PEL at all degrees of the ODP hierarchy as compared to UUP(e) and OSLo for any group size. The PaOSLo has 10% less average PEL as compared to UUP(e) and 2.5% less as compared to OSLo at degree 1 of the ODP hierarchy for the group size of three users. The PaOSLo has 9.6% and 4.7% less average PEL as compared to UUP(e) and OSLo at degree 1 of the ODP hierarchy for the group size of four users. Similarly, for the group size of five users, the PaOSLo has 8.7% and 4.36% better profile privacy as compared to UUP(e) and OSLo. Results show that PaOSLo provides better profile privacy as compared to UUP(e) and OSLo at all degrees of the ODP hierarchy for any group size. The reasons are the PaOSLo first cluster users based on their profile similarities. In such case, the group created by CS from different clusters contains users of dissimilar interest. Each user forwards a query of other users from the different clusters; hence, the profile of a user is obfuscated with the queries of users who have a dissimilar interest, and the user achieves maximum obfuscation.

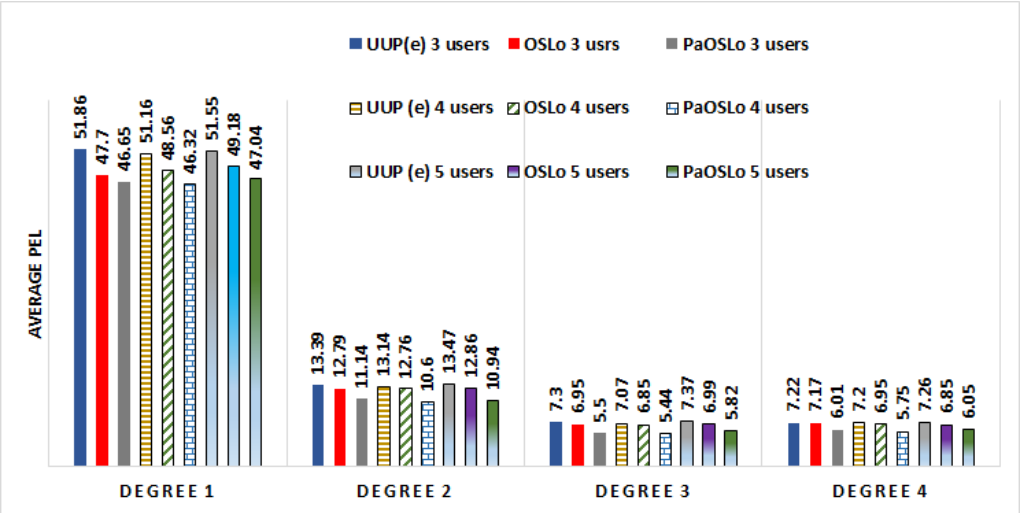

FIGURE 5.9: Average PEL of UUP(e) VS. OSLo VS. PaOSLo

5.4 Conclusion

The existing distributed privacy-preserving protocols create a group of 'n' users on first come first serve basis. Users are randomly grouped to forward each other's queries to the WSE without any prior knowledge of their interests. In random grouping, there is a greater chance that users having similar interests may be group together. Such grouping has a trivial effect on profile obfuscation. To overcome the limitation random grouping, this chapter proposes a novel privacy preserving protocol PaOSLo to obfuscate the profile with the queries of those users who have a dissimilar interest. Profile obfuscation with queries of dissimilar interest is the prime objective of PaOSLo. Profile building and finding the similarity between users' profiles are the primary steps of PaOSLo. The profile of the user is build using syntactic and semantic analysis of the query term, whereas, cosine similarity measures the resemblance between the users' profiles. K-Mean algorithm clusters the users' into three, four and five clusters based on the similarity computed in the previous step. To evaluate the impact of profile aware grouping, an experiment with the group size of three users, four users and five users is performed by selecting a user from each distinct cluster. A user executing PaOSLo forwards queries of other users of the group but not of his own query. The local privacy and profile privacy are the two dimensions to evaluate the privacy of a user.

The local privacy calculates the probability of linking the query to the user by a curious entity. As the user encrypts the query with the public key of SQFC, no

one will read the query content. However, If SQFC is curious and wants to find the query originator, $\frac{1}{(n-1)}$ is the probability of linking the query to the user. But if a SQFC makes a coalition with 'c' collaborators in a group $\frac{1}{(n-c)}$ is the probability of linking query with the user.

Profile privacy measures the extent of profile obfuscation by executing the PaOSLo. A privacy metric profile exposure level (PEL) measures the profile privacy of the user by calculating the difference between the original profile and obfuscated profile. An experiment is performed to compare the profile privacy a user achieves by executing PaOSLo and state-of-the-art privacy preserving protocol UUP(e) and OSLo based on the privacy metric PEL. The results show that when PaOSLo clustered the dataset into three clusters, it had preserved 10.04% and 2.2% better profile privacy at degree 1 of the ODP hierarchy for the group size of three users. Similarly, when the dataset is clustered into four groups and has executed the protocols with the group size of four users, PaOSLo preserved 9.46% and 4.61% better profile privacy as compared to UUP(e) and OSLo at degree 1 of the ODP hierarchy. Likewise, when the dataset is clustered into a count of five, the PaOSLo provides better profile privacy at degree 1 of the ODP hierarchy as compared to UUP(e) and OSLo. Furthermore, at higher degrees of the ODP hierarchy, PaOSLo has shown improved results as compared to UUP(e) and OSLo.

This chapter answer research question 2 of this dissertation mentioned in section 1.8. **Research Question 2.** *What is the effect of random grouping and profile aware grouping on the privacy of the user?*

This chapter answered the research question 2, the privacy a user achieved through profile-aware grouping is compared with the random grouping protocols (UUP(e) and OSLo). The results show that the profile-aware grouping provides better profile privacy as compared to random grouping. The result of profile-aware grouping is compared for degree 1 to degree 4 of the ODP hierarchy. Therefore, the concept of profile based grouping preserves better profile privacy as compared to random grouping. The user executing PaOSLo achieves the same local privacy as provided by OSLo, however, as improved profile privacy.

Chapter 6

Conclusion and Future Work

The Internet is an enormous warehouse of data, holding a range of material and containing information almost about everything. People from every category, class or country definitely need information residing in the WWW. The Web Search Engines (WSEs) like Google, Ask, Bing, AOL, Baidu etc., allow us to retrieve relevant information from the web using search queries. Through the searching and information retrieval process, the WSEs record all submitted queries called a query log. WSEs claim that they evaluate the query log through certain algorithms for a long period to profile and categorize the users according to the users' interests. The query log frequently holds sensitive data or information about the user submitting the query, and the dissemination of such data violates the user's privacy [15]. The release of such information poses a serious risk to the user privacy. Preserving web search privacy is the real concern of today's Internet life.

The existing techniques that preserve the Web search privacy of a user are classified into categories like standalone schemes, proxy servers / third party infrastructure, query scrabbling, hybrid techniques (encompassing standalone and proxy services) and distributed protocols. Unlinkability and indistinguishability are the two advantages of distributed protocols over other techniques. However, there are certain limitation in the existing distributed protocols, to tackle the limitation of existing distributed protocols. Three novel distributed protocol are proposed employing single dynamic groups named ObScure Logging (OSLo), Multi group distributed protocol termed as MG-OSLo, and a profile aware user grouping protocol called

PaOSLo. The objective of the OSLo and MG-OSLo is to provide both unlinkability (local privacy)and indistinguishability (profile privacy), whereas to obfuscate the profile with the users having dissimilar interest is the additional feature of PaOSLo. The section below describes the functionality and efficiency of each protocol.

6.1 ObScure Logging (OSLo)

To eliminate the limitation in the existing single group distributed privacy-preserving protocols and provide better privacy as compared to the state-of-the-art privacy-preserving protocol UUP(e) and co-utile, a novel single group privacy-preserving protocol OSLo is proposed to preserve the privacy of a user.

The local privacy and profile privacy are the two dimensions in which the privacy of a user were assessed. The query encryption, query shuffling, result encryption, and result broadcasting are the steps followed to enforce local privacy. The local privacy is measured using a probabilistic model relative to the entities involved in forwarding the query to the WSE. The probabilistic model illustrates that $\frac{1}{n-1}$ is the probability of linking the query to the user by SQFC. However, if the SQFC forms a coalition with other group users, $\frac{1}{n-C}$ is the probability of linking the query to the user where, c is the number of compromised users that have made a coalition with SQFC to link a query with the user.

To evaluate the profile privacy of a user, a comprehensive experimentation is performed with two datasets consisting of 500 users and 1000 users selected randomly from the AOL query log from the least active to highly active users. The experiment investigated the impact of the user count on profile privacy. The OSLo is simulated for two situations, i) self-query submission allowed, and ii) self-query submission not allowed. Results ensure that the average PEL drops by increasing the group size at all degrees of the ODP hierarchy.

The proposed protocol OSLo tackled the limitation mentioned in section 3.1, by achieving both local privacy and profile privacy. The local privacy acquired

through encryption and query shuffling. The encryption ensured the query content and the query result hidden from the group users, whereas, the query shuffling made the query unlinkable with the originating user. Hence, local privacy achieved both unlinkability and confidentiality. Additionally, each user has forwarded a variety of queries of other users resulted in the obfuscation of profile maintained by WSE. This chapter answered the first research question (RQ 1) along with its sub-parts RQ 1(a) and RQ 2(b). The key contribution of OSLo is that it offers, improved privacy in terms of unlinkability (local privacy) and indistinguishability (profile privacy) as compared to the state-of-the-art privacy-preserving protocols i.e., UUP(e) and the co-utile protocol. Furthermore, the impact of group sizes for a situation where self-query submission is allowed and self-query submission is not allowed are also investigated in chapter 3.

6.2 Multi Group ObScure Logging (MG-OSLo)

In the existing multi-group privacy-preserving protocols, memory locations were used to enforce privacy through unlinkability. A memory location is like a virtual box, a set of users associated with it could drop/write their query in it. Another user associated with the memory location is supposed to read the query and forward it to the WSE. A user who had access to the memory location can read the query contents and results in the query, hence compromising the privacy of the user. Further, the privacy of a user in a multi-group was only evaluated relative to the group users. To the best of our knowledge, that profile obfuscation was never evaluated in a multi-group protocol. To tackle the limitation in the existing multiple groups' privacy-preserving protocol, and to evaluate the profile privacy of a user along with privacy relative to the group users, a novel protocol called MG-OSLo is proposed. The local privacy and profile privacy are the two dimensions in which the privacy of a user are evaluated. To enforce the local privacy a non-overlapping group design (a user appears in a single group) and overlapping group design (a user appear in multiple groups) are considered to group the users. A probabilistic model calculates the probability that a curious entity has in linking the query to the user. In the case of non-overlapping group design, the probability

of linking the query to the user by a curious GSQFC is given by $\frac{1}{K(b-1)}$. The CS and group user cannot see the query or query result, as both are encrypted using an asymmetric key. However, if a curious CS builds a coalition with GSQFC, the probability of linking the query is $\frac{1}{K(b-1)}$. Similarly, if multiple GSQFC collaborate to find the query originating user, the probability of linking is $\frac{1}{K(b-C)}$. Even if all GSQFC are compromised including CS, the probability of linking query is 1/k.
In the case of overlapping group design, a balanced incomplete block design (BIBD) is considered to compute the local privacy (unlinkability) of a user. BIBD is based on (v, b, r, k, λ) configuration. In the overlapping design, two situation are considered, i) source and GSQFC belong to different groups. If the GSQFC is curious and wants to link query with the user, the probability of linking query is computed as $\frac{r}{b \cdot K)}$ ii) When source and GSQFC belong to the same group, two possible cases may arrive as a source and GSQFC paired in λ groups. In the first case when the GSQFC act as proxy U_j for source U_i and forwards the query to WSE, the probability of linking query with the user is computed as $\frac{1}{(K-1)}$. In the second case when the proxy $U_j \neq GSQFC$, the probability of linking query with the user is computed as $\frac{1}{\lambda(K-1)}$.

To measure the profile privacy of a user executing MG-OSLo, an experiment is performed to find the extent of profile obfuscation. The experiment consists of steps like simulating MG-OSLo for two situations (i.e., self-query submission allowed and self-query submission not allowed), building the profile for a user and computing the level obfuscation with the privacy metric PEL. The results for the first situation (allowing self query submission) show that the MG-OSLo provides better profile privacy and a lower PEL as compared to OSLo and co-utile for all group sizes when simulated with dataset 1. When the MG-OSLo is simulated with dataset 2, the MG-OSLo displayed improved privacy over OSLo as compared to OSLo and co-utile at degree 1 to degree 4 of ODP hierarchy.

Similarly, in the second situation, when self-query submission is not allowed, the profile privacy of MG-OSLo is compared with UUP(e) and OSLo. Results show that when these protocols were simulated for dataset 1 and dataset 2 for the group count of three, the MG-OSLo has also depicted better profile privacy as compared to UUP(e) and OSLo. The simulation results depict that a user achieved the

best profile privacy results with group count of three, each having three users in a group. In chapter 4, the MG-OSLo eliminated the limitation in the existing multi-group protocol. The research question (RQ 1) and its subparts (RQ 1(a) and RQ 1(b)) are addressed in chapter 4. The user achieved both the local privacy through encryption and query shuffling and profile privacy by forwarding the queries of the group users.

6.3 Profile Aware ObScure Logging (PaOSLo)

Group creation is the primary step of any distributed privacy-preserving protocols; each user in the group has to forward the query of other users to obfuscate the profile maintained by the WSE. In the existing protocols (OSLo, UUP(e)), a group of 'n' random users is created on a first come first serve basis. The random grouping approach can create a group of users having similar interests, making the profile insignificantly obfuscated. To create a group of a user having unrelated interests, a novel distributed privacy-preserving protocol, called profile aware ObScure Logging (PaOSLo), is proposed to achieve better obfuscation of the user's profile maintained by WSE. In the first step of PaOSLo execution, the users are clustered into groups based on their profile similarity before creating the group. The cosine similarity measure is used to compute the similarity between the users' profiles based on degree 1 of the ODP hierarchy. K-Mean algorithm clusters the users into three clusters, four clusters, and five clusters. Afterward, the CS created a group of users by selecting a user from each cluster, resulting in the profile of a user obfuscated with queries of those users having dissimilar interests. The impact of profile aware grouping on the profile privacy is evualted with dataset 2 in accordance with the step mentioned in section 3.5.2.

The Profile privacy a user achieved by executing PaOSLo is compared with the state-of-the-art privacy-preserving protocol UUP(e) and OSLo based on privacy metric PEL. The results showed that PaOSLo preserved 10.04% and 2.2% better profile privacy as compared to UUP(e) and OSLo for the group size of three users. Similarly, for the group size of four users, PaOSLo preserved 9.46% and 4.61% better profile privacy at degree 1 of the ODP hierarchy. Likewise, when five users

are grouped together, PaOSLo preserved 8.74% and 4.35% enhanced privacy as compared to UUP(e) and OSLo at degree 1 of the ODP hierarchy. The results show that at a higher degree of the ODP hierarchy PaOSLo provided better profile privacy as compared to UUP(e) and OSLo.

PaOSLo addressed the research question (RQ 2), it evaluated the impact of profile aware grouping vs random grouping. The results depicted that PaOSLo has a positive impact on profile privacy. A user achieved better profile privacy by grouping with those users having dissimilar interests as compared to the random grouping.

6.4 Limitation of Proposed Work

Web search privacy is an active area of research; people from academia, business, politics, etc. need to preserve their privacy when searching data via the WSE. However, there are some limitations associated with any distributed privacy-preserving protocols. The ethical issue is one of the major problems associated with a private web search. A user is possibly grouped with other users having no prior knowledge about their intention or preferences. It is likely that a user may forward an inappropriate query to the WSE on behalf of another user. Viejo et.al, in [43] have suggested the solution with regards to this ethical issue. Initially we have adopted the same solution to tackle this problem, however, in future alternatives are required to deal with this specific issue. The time required to create a group is a critical issue, although around 72 thousand queries are sent to Google in one second. According to the Poisson distribution, if there is an average of 72 queries per hundredth of a second, the probability of making a group of n=3 users to send 6 queries is 1 [13]. However, in realtime the number of users adopting distributed privacy-preserving protocol remains questionable. In this dissertation, the users that execute the proposed distributed protocols are considered curious but honest. However, if a user is selfish and forwards his query through other users but refuses to forward the queries of group users by leaving the group, it is another concern of distributed privacy-preserving protocols. Personalized results are more appealing to the user; profile obfuscation affects the quality of results. Poor personalized results are also an apprehension of distributed privacy-preserving protocols.

6.5 Future Work

In the future, the alternative way of user profiling needs to be investigated. The WSE keeps track of user profiles through many ways to provide personalized result. To enforce the web search privacy, these alternative ways of profiling require a further dimension of research. The effect of privacy on quality of results needs exploration. Moreover, how to get personalised results and privacy at the same time is a query for future exploration. The execution of distributed privacy-preserving protocol, such as encryption, shuffling, query forwarding, result processing and broadcasting, can cause significant delays in getting results from the WSE. The privacy and delay factor are another direction this research shall assess. Additionally, the privacy of the proposed distributed privacy-preserving protocols are compared with state-of-the-art distributed privacy-preserving protocol UUP(e)(e). However, the privacy of the proposed protocol shall be compared with the standalone scheme as well as the hybrid scheme.

Bibliography

[1] K. Purcell, L. Rainie, and J. Brenner, "Search engine use 2012," 2012. [Online]. Available: pewinternet.org/Reports/2012/Search-Engine-Use-2012.aspx.

[2] "Internet live stats - internet usage social media statistics," last accessed June 16, 2019. [Online]. Available: https://www.internetlivestats.com/

[3] O. Y. Rieger, "Search engine use behavior of students and faculty: User perceptions and implications for future research," *First Monday*, vol. 14, no. 12, 2009.

[4] O. Dan and B. D. Davison, "Measuring and predicting search engine users' satisfaction," *ACM Computing Surveys (CSUR)*, vol. 49, no. 1, pp. 1–35, 2016.

[5] N. Kaaniche, S. Masmoudi, S. Znina, M. Laurent, and L. Demir, "Privacy preserving cooperative computation for personalized web search applications," in *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, 2020, pp. 250–258.

[6] A. Hannak, P. Sapiezynski, A. Molavi Kakhki, B. Krishnamurthy, D. Lazer, A. Mislove, and C. Wilson, "Measuring personalization of web search," in *Proceedings of the 22nd international conference on World Wide Web*. ACM, 2013, pp. 527–538.

[7] C. Romero-Tris, J. Castella-Roca, and A. Viejo, "Distributed system for private web search with untrusted partners," *Computer Networks*, vol. 67, pp. 26–42, 2014.

[8] J. Parra-Arnau, D. Rebollo-Monedero, and J. Forné, "Measuring the privacy of user profiles in personalized information systems," *Future Generation Computer Systems*, vol. 33, pp. 53–63, 2014.

[9] A. Cooper, "A survey of query log privacy-enhancing techniques from a policy perspective," *ACM Transactions on the Web (TWEB)*, vol. 2, no. 4, pp. 1–27, 2008.

[10] C. Wei, Q. Gu, S. Ji, W. Chen, Z. Wang, and R. Beyah, "Ob-wspes: A uniform evaluati on system for obfuscation-based web search privacy," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–18, 2019.

[11] F. Saint-Jean, A. Johnson, D. Boneh, and J. Feigenbaum, "Private web search," in *Proceedings of the 2007 ACM workshop on Privacy in electronic society*. ACM, 2007, pp. 84–90.

[12] B. C. Fung, K. Wang, A. W.-C. Fu, and S. Y. Philip, *Introduction to privacy-preserving data publishing: Concepts and techniques*. Chapman and Hall/CRC, 2010.

[13] M. Ullah, M. A. Islam, R. Khan, M. Aleem, and M. A. Iqbal, "Obsecure logging (oslo): A framework to protect and evaluate the web search privacy in health care domain," *Journal of Medical Imaging and Health Informatics*, vol. 9, no. 6, pp. 1181–1190, 2019.

[14] Mar 2013 (accessed May 15, 2019). [Online]. Available: http://www.google.com/privacy

[15] P. Eckersley, "How unique is your web browser?" in *International Symposium on Privacy Enhancing Technologies Symposium*. Springer, 2010, pp. 1–18.

[16] S. Englehardt, D. Reisman, C. Eubank, P. Zimmerman, J. Mayer, A. Narayanan, and E. W. Felten, "Cookies that give you away: The surveillance implications of web tracking," in *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2015, pp. 289–299.

[17] G. Acar, M. Juarez, N. Nikiforakis, C. Diaz, S. Gürses, F. Piessens, and B. Preneel, "Fpdetective: dusting the web for fingerprinters," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. ACM, 2013, pp. 1129–1140.

[18] N. Nikiforakis, A. Kapravelos, W. Joosen, C. Kruegel, F. Piessens, and G. Vigna, "Cookieless monster: Exploring the ecosystem of web-based device fingerprinting," pp. 541–555, 2013.

[19] D. Pàmies-Estrems, J. Castellà-Roca, and J. Garcia-Alfaro, "A real-time query log protection method for web search engines," *IEEE Access*, vol. 8, pp. 87 393–87 413, 2020.

[20] A. J. Biega, R. Saha Roy, and G. Weikum, "Privacy through solidarity: A user-utility-preserving framework to counter profiling," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2017, pp. 675–684.

[21] H. Wang, W. Liu, and J. Wang, "Achieve web search privacy by obfuscation," in *International Conference on Security with Intelligent Computing and Big-data Services*. Springer, 2019, pp. 315–328.

[22] M. Barbaro, T. Zeller, and S. Hansell, "A face is exposed for aol searcher no. 4417749," *New York Times*, vol. 9, no. 2008, pp. 1–8, 2006.

[23] A. Arampatzis, G. Drosatos, and P. S. Efraimidis, "Versatile query scrambling for private web search," *Information Retrieval Journal*, vol. 18, no. 4, pp. 331–358, 2015.

[24] R. Khan, M. A. Islam *et al.*, "Quantification of pir protocols privacy," in *2017 International Conference on Communication, Computing and Digital Systems (C-CODE)*. IEEE, 2017, pp. 90–95.

[25] R. Khan, M. Ullah, and M. A. Islam, "Revealing pir protocols protected users," in *2016 Sixth International Conference on Innovative Computing Technology (INTECH)*. IEEE, 2016, pp. 535–541.

[26] M. Rasch, "Google's data minefield," *The Register*, 2006, last accessed December 20, 2018. [Online]. Available: http://www.theregister.co.uk/2006/01/31/googlesubpoenausgovernment/

[27] R. Khan, A. Ahmad, A. O. Alsayed, M. Binsawad, M. A. Islam, and M. Ullah, "Qupid attack: Machine learning-based privacy quantification mechanism for pir protocols in health-related web search," *Scientific Programming*, vol. 2020, p. DOI: https://doi.org/10.1155/2020/8868686, 2020.

[28] K. Hafner and M. Richtel, "Google resists us subpoena of search data," *New York Times*, vol. 20, pp. 1–3, 2006.

[29] S. T. Peddinti and N. Saxena, "Web search query privacy: Evaluating query obfuscation and anonymizing networks 1," *Journal of Computer Security*, vol. 22, no. 1, pp. 155–199, 2014.

[30] J. Yang, M. M. H. Onik, N.-Y. Lee, M. Ahmed, and C.-S. Kim, "Proof-of-familiarity: A privacy-preserved blockchain scheme for collaborative medical decision-making," *Applied Sciences*, vol. 9, no. 7, pp. 1370–1394, 2019.

[31] K. Mathews-Hunt, "Cookieconsumer: Tracking online behavioural advertising in australia," *Computer Law & Security Review*, vol. 32, no. 1, pp. 55–90, 2016.

[32] R. Esguerra, "Google ceo eric schmidt dismisses the importance of privacy," *Electronic Frontier Foundation*, vol. 10, 2009, last accessed March 22, 2019. [Online]. Available: https://www.eff.org/deeplinks/2009/12/google-ceo-eric-schmidt-dismisses-privacy/

[33] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," Naval Research Lab Washington DC, Tech. Rep., 2004.

[34] A. Petit, T. Cerqueus, S. B. Mokhtar, L. Brunie, and H. Kosch, "Peas: Private, efficient and accurate web search," in *2015 IEEE Trustcom/BigDataSE/ISPA*, vol. 1. IEEE, 2015, pp. 571–580.

[35] S. B. Mokhtar, A. Boutet, P. Felber, M. Pasin, R. Pires, and V. Schiavoni, "X-search: revisiting private web search using intel sgx," in *Proceedings of the 18th ACM/IFIP/USENIX Middleware Conference*. ACM, 2017, pp. 198–208.

[36] A. Arampatzis, P. Efraimidis, and G. Drosatos, "Enhancing deniability against query-logs," in *European Conference on Information Retrieval*. Springer, 2011, pp. 117–128.

[37] A. Petit, T. Cerqueus, A. Boutet, S. B. Mokhtar, D. Coquil, L. Brunie, and H. Kosch, "Simattack: private web search under fire," *Journal of Internet Services and Applications*, vol. 7, no. 1, pp. 1–17, 2016.

[38] M. K. Reiter and A. D. Rubin, "Crowds: Anonymity for web transactions," *ACM transactions on information and system security (TISSEC)*, vol. 1, no. 1, pp. 66–92, 1998.

[39] J. Domingo-Ferrer, M. Bras-Amorós, Q. Wu, and J. Manjón, "User-private information retrieval based on a peer-to-peer community," *Data & Knowledge Engineering*, vol. 68, no. 11, pp. 1237–1252, 2009.

[40] J. Castellà-Roca, A. Viejo, and J. Herrera-Joancomartí, "Preserving user's privacy in web search engines," *Computer Communications*, vol. 32, no. 13-14, pp. 1541–1551, 2009.

[41] C. M. Swanson and D. R. Stinson, "Extended combinatorial constructions for peer-to-peer user-private information retrieval," *arXiv preprint arXiv:1112.2762*, 2011.

[42] C. Swanson and D. R. Stinson, "Extended results on privacy against coalitions of users in user-private information retrieval protocols," *Cryptography and Communications*, vol. 7, no. 4, pp. 415–437, 2015.

[43] A. Viejo and J. Castellà-Roca, "Using social networks to distort users' profiles generated by web search engines," *Computer Networks*, vol. 54, no. 9, pp. 1343–1357, 2010.

[44] A. Erola, J. Castellà-Roca, A. Viejo, and J. M. Mateo-Sanz, "Exploiting social networks to provide privacy in personalized web search," *Journal of Systems and Software*, vol. 84, no. 10, pp. 1734–1745, 2011.

[45] J. Domingo-Ferrer, S. Martínez, D. Sánchez, and J. Soria-Comas, "Co-utile p2p anonymous keyword search," in *Co-utility*. Springer, 2018, pp. 51–70.

[46] A. Pfitzmann and M. Köhntopp, "Anonymity, unobservability, and pseudonymity—a proposal for terminology," in *Designing privacy enhancing technologies*. Springer, 2001, pp. 1–9.

[47] S. Brier, "How to keep your privacy: Battle lines get clearer," *The New York Times*, vol. 13, January 1997, last accessed on May 16, 2019. [Online]. Available: www.nytimes.com/1997/01/13/business/how-to-keep-yourprivacy-battle-lines-get-clearer.html

[48] D. C. Howe and H. Nissenbaum, "Trackmenot: Resisting surveillance in web search," *Lessons from the Identity trail: Anonymity, privacy, and identity in a networked society*, vol. 23, pp. 417–436, 2009.

[49] S. T. Peddinti and N. Saxena, "On the privacy of web search based on query obfuscation: a case study of trackmenot," in *International Symposium on Privacy Enhancing Technologies Symposium*. Springer, 2010, pp. 19–37.

[50] J. Domingo-Ferrer, A. Solanas, and J. Castellà-Roca, "h (k)-private information retrieval from privacy-uncooperative queryable databases," *Online Information Review*, vol. 33, no. 4, pp. 720–744, 2009.

[51] M. Juarez and V. Torra, "Dispa: An intelligent agent for private web search," in *Advanced Research in Data Privacy*. Springer, 2015, pp. 389–405.

[52] G. Weinberg, "Privacy, simplified." 2008, last accessed on Feb 17, 2019. [Online]. Available: https://duckduckgo.com/

[53] P. Bradley, "Search engines:'ixquick', a multi-search engine with a difference," *Ariadne*, vol. 23, 2000.

[54] A. Raza, K. Han, and S. O. Hwang, "A framework for privacy preserving, distributed search engine using topology of dlt and onion routing," *IEEE Access*, vol. 8, pp. 43 001–43 012, 2020.

[55] R. Pires, D. Goltzsche, S. B. Mokhtar, S. Bouchenak, A. Boutet, P. Felber, R. Kapitza, M. Pasin, and V. Schiavoni, "Cyclosa: Decentralizing private web search through sgx-based browser extensions," in *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2018, pp. 467–477.

[56] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, "Private information retrieval," in *Proceedings of IEEE 36th Annual Foundations of Computer Science*. IEEE, 1995, pp. 41–50.

[57] E. Kushilevitz and R. Ostrovsky, "Replication is not needed: Single database, computationally-private information retrieval," in *Proceedings 38th Annual Symposium on Foundations of Computer Science*. IEEE, 1997, pp. 364–373.

[58] R. Ostrovsky and W. E. Skeith, "A survey of single-database private information retrieval: Techniques and applications," in *International Workshop on Public Key Cryptography*. Springer, 2007, pp. 393–411.

[59] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE transactions on information theory*, vol. 31, no. 4, pp. 469–472, 1985.

[60] Y. Lindel l and E. Waisbard, "Private web search with malicious adversaries," in *International Symposium on Privacy Enhancing Technologies Symposium*. Springer, 2010, pp. 220–235.

[61] D. Chaum, "Untraceable electronic mail, return addresses and digital pseudonyms," in *Secure electronic voting*. Springer, 2003, pp. 211–219.

[62] R. Cramer, G. Hanaoka, D. Hofheinz, H. Imai, E. Kiltz, R. Pass, A. Shelat, and V. Vaikuntanathan, "Bounded cca2-secure encryption," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2007, pp. 502–518.

[63] Z. Cao, L. Liu, and Z. Yan, "An improved lindell-waisbard private web search scheme." *IJ Network Security*, vol. 18, no. 3, pp. 538–543, 2016.

[64] M. K. Wright, M. Adler, B. N. Levine, and C. Shields, "The predecessor attack: An analysis of a threat to anonymous communications systems," *ACM Transactions on Information and System Security (TISSEC)*, vol. 7, no. 4, pp. 489–522, 2004.

[65] G. Navarro-Arribas, *Advanced Research in Data Privacy*. Springer International PU, 2016, vol. 567.

[66] C. Romero-Tris, A. Viejo, and J. Castellà-Roca, "Multi-party methods for privacy-preserving web search: Survey and contributions," in *Advanced Research in Data Privacy*. Springer, 2015, pp. 367–387.

[67] K. Stokes and M. Bras-Amorós, "Optimal configurations for peer-to-peer user-private information retrieval," *Computers & mathematics with applications*, vol. 59, no. 4, pp. 1568–1577, 2010.

[68] N. Kaaniche, S. Masmoudi, S. Znina, M. Laurent, and L. Demir, "Privacy preserving cooperative computation for personalized web search applications," in *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, 2020, pp. 250–258.

[69] J. Domingo-Ferrer, A. Viejo, F. Sebé, and Ú. González-Nicolás, "Privacy homomorphisms for social networks with private relationships," *Computer Networks*, vol. 52, no. 15, pp. 3007–3016, 2008.

[70] J. Karmeshu, *Entropy measures, maximum entropy principle and emerging applications*. Springer Science & Business Media, 2003, vol. 119.

[71] C. Diaz, S. Seys, J. Claessens, and B. Preneel, "Towards measuring anonymity," in *International Workshop on Privacy Enhancing Technologies*. Springer, 2002, pp. 54–68.

[72] M. Ullah, R. Khan, and M. A. Islam, "Poshida, a protocol for private information retrieval," in *2016 Sixth International Conference on Innovative Computing Technology (INTECH)*. IEEE, 2016, pp. 464–470.

[73] U. Mohib, R. Khan, and M. A. Islam, "Poshida ii, a multi group distributed peer to peer protocol for private web search," in *2016 International Conference on Frontiers of Information Technology (FIT)*. IEEE, 2016, pp. 75–80.

[74] A. Waksman, "A permutation network," *Journal of the ACM (JACM)*, vol. 15, no. 1, pp. 159–163, 1968.

[75] C. Carpineto and G. Romano, "A review of ten year research on query log privacy." in *7th Italian Information Retrieval Workshop*, 2016.

[76] A. Kumar, *Web Usage Mining Techniques and Applications Across Industries*. IGI Global, 2016.

[77] M. Kamvar and S. Baluja, "A large scale study of wireless search behavior: Google mobile search," in *Proceedings of the SIGCHI conference on Human Factors in computing systems*, 2006, pp. 701–709.

[78] P. Arora, A. Singh, and H. Tyagi, "Evaluation and comparison of security issues on cloud computing environment," *World of Computer Science and Information Technology Journal (WCSIT)*, vol. 2, no. 5, pp. 179–183, 2012.

[79] G. Singh, "A study of encryption algorithms (rsa, des, 3des and aes) for information security," *International Journal of Computer Applications*, vol. 67, no. 19, pp. 33–38, 2013.

[80] A. Siddharthan, "Christopher d. manning and hinrich schutze. foundations of statistical natural language processing. mit press, 2000. isbn 0-262-13360-1. 620 pp. $64.95£ 44.95 (cloth)." *Natural Language Engineering*, vol. 8, no. 1, pp. 91–92, 2002.

[81] N. Senthilkumar and P. R. Ch, "Prediction of user interest fluctuation using fuzzy neural networks in web search," *International Journal of Intelligent Unmanned Systems*, 2020.

[82] DMOZ, "Odp, open directory project," 2013 (accessed March 15, 2017). [Online]. Available: http://www.dmoz.org/

[83] A. Viejo, J. Castella-Roca, O. Bernadó, and J. M. Mateo-Sanz, "Single-party private web search," in *2012 Tenth Annual International Conference on Privacy, Security and Trust*. IEEE, 2012, pp. 1–8.

[84] G. Sterling, M. Beck, and A. Gesenhues, "Stats: comscore archives," accessed on May 27, 2019. [Online]. Available: https://searchengineland.com/library/stats/stats-comscore

[85] K. Stokes and M. Bras-Amorós, "On query self-submission in peer-to-peer user-private information retrieval," in *Proceedings of the 4th International Workshop on Privacy and Anonymity in the Information Society*. ACM, 2011, p. 7.

[86] A. Dey, *Theory of block designs*. J. Wiley, 1986.

[87] A. Huang, "Similarity measures for text document clustering," in *Proceedings of the sixth new zealand computer science research student conference (NZCSRSC2008), Christchurch, New Zealand*, vol. 4, 2008, pp. 9–56.

[88] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *ACM SIGKDD explorations newsletter*, vol. 11, no. 1, pp. 10–18, 2009.

[89] P. Bhatia, *Data Mining and Data Warehousing: Principles and Practical Techniques*. Cambridge University Press, 2019.

[90] J. Wu, *Advances in K-means clustering: a data mining thinking*. Springer Science & Business Media, 2012.