iShield: A Framework for Preserving Privacy of iOS App User

Arpita Jadhav Bhatt*, Chetna Gupta and Sangeeta Mittal

Department of Computer Science & IT
Jaypee Institute of Information Technology, Noida, India
E-mail: arpitajadhav@gmail.com; arpita.jadhav@jiit.ac.in;
chetna.gupta@jiit.ac.in; sangeeta.mittal@jiit.ac.in
**Corresponding Author*

Received 21 January 2019; Accepted 31 July 2019;
Publication 12 August 2019

Abstract

Do iOS apps honour user's privacy? Protection of user's privacy by apps has lately emerged as a big challenge. Many studies have identified that there exists an inherent trade-off between end user's privacy and apps' functionality. Some methods have been proposed to preserve user's privacy of specific data like location and health information. However, a comprehensive framework to enable privacy preserving data sharing by apps has not been found. In this paper, we have proposed *iShield* - a privacy preserving framework that can be easily integrated by developers at the time of app creation to enforce privacy with minimal performance overhead. Privacy threat to a user has been quantified by calculation of privacy disclosure score of an app user. Empirical results demonstrate that the approach significantly reduces the privacy disclosure of the user.

Keywords: Privacy preserving framework, iOS Apps, static and dynamic analysis, information security.

Journal of Cyber Security and Mobility, Vol. 8_4, 493–536.
doi: 10.13052/jcsm2245-1439.845

1 Introduction

With the increasing availability and demand of Smartphones, there is growing demand for feature rich applications [1]. These include multimedia, gaming, live data streaming, location based, education, instant messaging and social networking apps. To provide users with functionality, apps often require access to potentially sensitive user data like address book, photo album and sensor-based data like location, camera, microphone etc. For example, a speech recognition app needs access to recordings of user's speech, microphone etc. Most of the apps use this sensitive data responsibly, but there have been several evidences of privacy violations by third-party apps [2]. According to a recent report by Appthority, thousands of iOS and Android apps have collectively leaked data of millions of users [3].

Existing studies lack in proposing methods to preserve entire data of user that is shared over the network by the apps. Permission based access mechanisms has not proven to be effective. Many users are inundated with permission requests and may not completely understand its repercussions after indiscriminately granting all requests or may disable notifications completely entrusting all apps with their private and sensitive data. Moreover, permissions are very coarse grained. For example, an app that has permission to access network can send any type of network information to several domains [2]. Moreover, studies have revealed that apps often request more permissions than they require and may use granted permissions in unexpected ways [2]. For example, an app with location permission may transmit location information such as geo-coordinates, zip code, address, street name etc. to other third parties [1]. Other private and sensitive user's data may include live photo sharing, live sharing of recorded audio, clinical data, bank credentials, email and passwords etc. Apart from containing user permissions-based access, developers must also preserve user's privacy before sharing sensitive data over the network after the user has granted permission to an app.

There is a dearth of tools or frameworks to accomplish total privacy. There exists lack of control for the users to limit the apps in sharing their data [2], thus it becomes the responsibility of app developers to protect user's data. However, past studies have identified that most of the app developers (i) do not apply encryption techniques before sharing user's data over the network [3–6] and (ii) often create privacy infringing apps or over privileged apps. Therefore, cost-effective techniques are required to preserve user's privacy and fulfill apps functional requirements (via permissions). To bridge this gap, iShield

framework has been proposed which can be integrated easily like third-party framework. iShield uses data perturbation and data encryption techniques appropriately to shield sensitive data from being exposed to perpetrators. Framework can be imported by app developers to enhance privacy quotient of the apps. Effectiveness of iShield framework to preserve end user's privacy has been demonstrated with the calculation of privacy metric, *privacy disclosure score* before and after applying privacy preserving strategies. Results demonstrate that the proposed approach enhances the privacy level of a user by 57.72%.

1.1 Motivation of Creating iShield Framework

To enforce security and protect iOS devices, Apple adopts application sandboxing [7]. Application sandboxing also called as application containerization, is an approach for mobile application management (MAM) and software development, which limits the environment in which a piece of code can execute [8]. The primary goal of application sandboxing is improving the security of an application by isolating it and preventing it from intruders, outside malware attacks, system resources or any other applications, from interacting with the protected application [7, 9]. Every app is given a sandbox and a directory to store data in it. If an app needs to access data that is not located in its sandbox, then it needs to request data through system interface. System interface adds a layer of security, as the operating system knows what data app is accessing. Thus, due to application sandboxing, apps cannot access other apps data. Therefore, designing an app that classifies behavior of other installed privacy infringing apps is not possible as it is against the design policy of application sandboxing. Such type of application can be developed at OS (operating system) level and by developers of iOS (iPhone operating system) and not at application level by third-party developers. Moreover, our past studies on network forensic analysis of 20 iOS apps have identified that 95% of apps share authorized or unauthorized data of app user in unencrypted form [6]. For 95% of the apps we were able to reconstruct partial or complete user's data on different workstations. Therefore, we propose iShield framework which can be integrated by app developers (at the time of app creation) like any other third-party framework to enforce privacy. Though, iShield framework can result in minimal increased execution time of app, but is capable of preserving user's privacy. iShield framework incorporates data perturbation techniques to perform series of simple operations and replace sensitive data by modified values. The motivation of using data perturbation techniques instead of data

Table 1 Performance evaluation of data perturbation and data encryption

Parameters	Data Perturbation	Data Encryption
Privacy preserving scale	Medium	High
Computing cost	Low	High
Communication overhead deployment complexity	Low	High

encryption techniques is its low computation overhead. Table 1 depicts the performance evaluation of both the techniques [10].

Apart from the above parameters data perturbation techniques are keyless whereas data encryption whether symmetric or asymmetric requires generation of keys and their maintenance. If the keys are lost, data associated with the key is also lost [11, 12]. There are other limitations with data encryption such as speed, certification problems. We therefore, use data perturbation techniques (detailed in Section 5) as it has low computing cost and low communication overhead deployment complexity. Following section describes the related work.

The major contributions of the work are summarized below:

- Model privacy threats of an app user
- Quantify privacy risks as privacy disclosure score
- Propose a privacy preserving framework that can be easily integrated by the app developers like any other third-party framework.
- Demonstrate the proposed concepts by examples and case studies

The rest of the paper is organized as follows: Section 2 discusses related work followed by Section 3 that describes overall methodology. Section 4 describes implementation of iShield framework. Section 5 details the implementation of privacy preserving methods of iShield. Section 6 describes the performance evaluation of iShield framework. The paper is concluded in Section 7.

2 Related Work

The word privacy has various definitions as it can vary from ‘information privacy’ to ‘personal privacy’ [13]. Our study is principally concerned with protecting user’s data privacy. Researchers of Proof-point [14] have analyzed several categories of free apps that are hosted on official stores of Android and iOS and have found that these shared tons of users’ private data. The study included examples of several friendly apps such as Bible app, flashlight app which contained malicious code. Some apps were capable of reading

user's SMS messages, others traced GPS location and many communicated data to numerous servers across many countries [14]. Rene Millman analyzed 110 apps and found that 73% Android and 47% iOS apps leaked personal data to third party domains [4]. Apart from the above there are several reports that highlight how these mobile apps share user's information over the network [15], whereas some highlight that the developers of the mobile apps do not consider user's privacy as an important parameter while sharing data over the network by intentionally/unintentionally creating privacy infringing apps or over privileged apps demanding extra permissions [5]. Therefore, one of the biggest challenges is to secure user's private and sensitive information. Given the increasing concerns about user's privacy many government organizations and enterprises are initiating regulations, for example, US government has initiated location privacy act [1]. US Congress has given due consideration to privacy of user's browser history and enforced that service providers must get consumers consent before using browsing history for advertisement and marketing purpose [16]. There are various researches conducted on privacy concerns for online networks including social networks that deal with publishing user's data without leaking identity of user. Yet there has been minimal attention towards privacy from user's perspective i.e. risks which arise due to information sharing over the network. In this context, we are concerned with user's information that is shared by mobile apps after user runs the application and grants permission to the app. Several methods and techniques have been proposed to compute privacy and information sharing in public domains, including statistical and algorithmic approaches. Fang et al. [17] have provided a survey report on privacy preserving techniques in big data and its key techniques. The authors have discussed the concepts of big data, privacy preserving techniques, and their challenges in protecting personal privacy. The authors have also emphasized that there must be strict laws and regulations for solving big data privacy preserving problem. Bertino et al. [18] have provided comprehensive views on metrics related with the existing privacy preserving algorithms. Their study provides insights to the developers/users/enterprises to choose and design privacy preserving data mining algorithms and apply privacy effective measures. The authors have reviewed and summarized existing privacy preserving criteria and metrics for evaluating privacy preserving techniques. Mendes et al. [19] provide a survey on most relevant privacy preserving data mining (PPDM) techniques for existing literature along-with the metrics that are used to evaluate the privacy preserving data mining techniques. They have also presented the application of PPDM methods

for different fields such as location based, cloud, e-Health etc. The authors describe PPDM methods according to their different phases such as collection of data, publishing data, distribution and output of data. Aghasian et al. [13] have proposed an approach that helps the social media users to assess their privacy disclosure score based on information shared across multiple online social networking sites. Authors have identified the main factors that impact user's privacy such as sensitivity, visibility to compute final disclosure score for every user. The authors have also applied concepts of statistical and fuzzy system to specify potential information loss using privacy disclosure score. The advantage of their proposed approach is that they provide better estimation of privacy disclosure score across multiple online social networks. Though, authors have compiled the information disclosure of an individual user from multiple sources; their study lacks methods or techniques that can reduce privacy disclosure score of an individual. Gao et al. [20] propose a novel method of protecting location privacy called as Hidden Ring and Hidden Forest. The technique uses breadth first search to meet requirements of ring and forest. The authors have tested technique on real and simulated data sets, and have demonstrated the effectiveness of method in protecting location privacy. Although, the proposed approach is effective in protecting user's location; the technique is not applicable for all types' permission like camera and data shared by camera application or calendar permission etc. Saranya et al. [21] have analyzed use of normalization techniques to achieve privacy. The authors have applied three normalization technique Min-max normalization, Z-score and Decimal scale normalization and have concluded that Min-max normalization technique outperforms the other approaches and it has minimum misclassification error. In our work we have applied Min-Max normalization techniques for perturbing numeric data (detailed in Section 6). Sy et al. [22] have proposed an infrastructure that simplifies use of privacy enhancing technique in apps. The proposed framework provides incentive for application developers to create privacy preserving apps. Their framework supports various type of anonymity which is relevant for real-world use cases. Liu et al. have proposed human centric privacy measuring framework PriMe [23] that quantifies privacy risks based on user's preference towards sharing data in mobile participating sensing systems. The authors have implemented and deployed PriMe as real time application for user study and evaluation. However, their study lacks in protecting user's data because once the user grants permission to an app, they never know how the data associated with a given permission is shared. For example, if a user allows an application to access his/her location, it is difficult for the user to know

how frequently the application is sharing this data once the permission has been granted. Benign apps prevent user's data as most of the information that is shared is encrypted, however there may be malicious apps that may publish/leak user's data. Therefore, we propose iShield framework, which can be integrated like any third-party framework to enforce privacy by use of privacy preserving techniques. The methods defined in the frameworks are reusable and can be extended in accordance with features of applications. iShield decreases the privacy disclosure of the users by applying techniques that aim to reduce the visibility of the attributes that are shared over network. Our work is well demonstrated with the help of numerical calculations, privacy preserving algorithms. With respect to privacy preserving techniques, data reconstruction techniques that help the receivers to reconstruct original data have also been proposed.

3 Methodology

The overall approach adopted in this work is depicted in Figure 1. End user installs apps from App Store or other third-party stores. When user runs the app, it requests permissions to access resources like Bluetooth, location, calendar, photos and camera. On running, the app shares a lot of user's data over the network. These include user's data with respect to permission like latitude/longitude via location permission, images via camera permission, details of events via calendar etc. Apart from data associated with permissions, iOS apps can also share device related data and user's personally identifiable information which are detailed in Figure 1.

Many background studies and our previous work on dynamic analysis has revealed that most of the apps share user's data in clear text over the network [24]. The privacy concerns get multiplexed because the permissions are coarsely-grained. For example, an app having location permission can share each and every minute details of users including postal code, apartment name etc. Apart from the above problem, users cannot limit the apps when they share user's data to their own domain or with third-party domain resulting in privacy breaches [2].

Therefore, to handle the above problems, iShield framework is proposed which is designed to protect end user's privacy by implementing privacy preserving approaches. iShield incorporates various data perturbation techniques based on the data-types of user attributes shared due to permissions. Subsequent section describes the threat model.

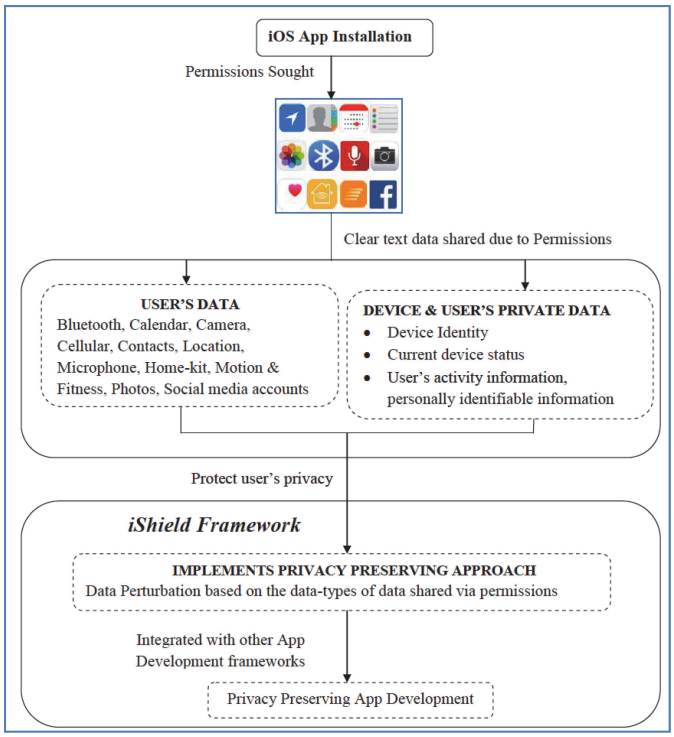

Figure 1 Methodology.

3.1 Threat Model

Authors have developed models of privacy invasiveness by apps and their detection at run time as well as before installation in [24] and [25]. Thus, it is assumed that only the apps from non-malicious publishers have been installed. More restrictions in this phase of analysis would leave out some useful apps too.

iShield is a privacy preserving framework means for use in apps that pass the static and dynamic privacy breach analysis suggested in [24] and [25]. It can be used for balancing app functionalities and user's privacy. Thus, our threat model assumes that app developers do not have any intention of leaking user's data in transit. There are three entities involved in communication [2] (i) platform provider that provides s third-party apps, hardware and OS software. iPhone/iPad/iPod are some platforms provided by Apple (ii) Users, who install and run trusted/untrusted apps on their iOS devices and (iii) publishers provide apps on Apple's app store or any other third-party stores.

Publishers use third-party frameworks for crash-reporting, data analytics and advertisements. These frameworks may send user's data in clear text, even without the knowledge of publisher, thereby raising privacy concerns. So, it is considered that there is a threat from random eavesdropper who may be seeking to intercept ongoing session between user and app server. iShield framework protects against this threat of possible private data harvesting by the eavesdropper. After including iShield, the publisher can use any other third-party framework without being concerned about privacy threats. Like any other software, if techniques proposed are implemented incorrectly, it may be subjected to code injection or other network attacks that may compromise user's security [2]. These attacks are outside the scope of this paper and the research work primarily focuses on privacy preserving approaches for each possible data type in app.

3.2 Privacy Preserving Approach of iShield

The proposed strategies in this work intend to enhance privacy of user by preventing unauthorized access to data in transit. iShield is designed to handle all types of user permissions and their attributes. In this work, we have considered 13 resource permissions (detailed in Table 2) that can leak user's private data. Techniques have been proposed to protect data transmitted due to these permissions. The proposed privacy preserving strategies have been applied on real time data captured while the app was running (detailed

Table 2 User permissions and their risk

Permission	Risk
Bluetooth	Collecting information on connected devices, types of file shared
Calendar	Collecting information on user's events, tasks, notes & reminders
Camera	Activating camera, live images, recorded videos
Cellular data	Collecting cellular information, remote numbers connected on call, internet connectivity, call duration
Address Book	Collecting user's contact list, contact's personal information
Location	Collecting user's geo-location
Microphone	Collecting audio files, live audio
HomeKit	Collecting information on rooms and devices which are configured, identifying a home address
HealthKit	Collecting disease information, symptoms
Motion & Fitness	Processing Medical diagnosis
Reminders	Collecting textual information
Social Accounts	Collecting account credentials, passwords, tweets, posts, location tags on photos
Photos	Collecting live images

in Section 3). These techniques have been evaluated with the help of privacy metric: privacy disclosure score to demonstrate the effectiveness of the applied approach in protecting user's privacy (detailed in Section 4).

4 Implementation of Privacy Preserving iShield Framework

Privacy refers to sensitive information or data that individuals, enterprises or organizations do not want to be known or disclosed to outside world [18]. Examples can include salary, emails, password, patient records, or a company's financial information. However, exact definition of privacy may vary between data owners. For example, some users may feel uploading photos with geo-tagging as personal privacy, whereas an open-minded person may not feel so. For protecting user's privacy several efforts are done by incorporating privacy preserving techniques to prevent disclosure of user's sensitive information when their data is shared over network. The existing privacy preserving techniques in data mining are classified in five different dimensions [18]: (i) data distribution (ii) modification using encryption, generalization, and perturbation etc., (iii) data mining algorithm for privacy preservation, (iv) data type that needs protection from disclosure, (v) adopted approach for preserving privacy i.e. heuristic or cryptography. In this work, we propose data type specific privacy enhancing techniques with privacy disclosure score as privacy metric. Model for privacy leaks and privacy preservation techniques has been discussed in separate subsections here.

4.1 Permission Induced Risks for Mobile Users

Both iOS and Android mobile operating systems rely on permission-based access control declaration mechanisms. For both operating systems, applications must request permissions to be allowed to access user's/device sensitive resources. Examples include address book, location, photo gallery, camera, Bluetooth, microphone etc. Though, user permissions notify the resources that an application attempts to access, but they fail to provide fine-grained information on how and when these resources will be used. In this work, we analyzed privacy risks for mobile iOS users based on permissions (refer Table 2 [16]).

Our previous works have identified that apart from permissions related data, mobile apps also leak user's personally identifiable information such as passwords, credit card numbers, banking information, email content, medical data, profile images, multimedia contents such as videos, audio, photos, and sleep pattern etc. as well as device related information such as name of Wi-Fi,

MAC address, device version, model number etc. [24]. Thus, in this phase several privacy preserving techniques have been proposed to protect entire data of the users that is shared across the network. Following subsection describes the proposed approach to calculate privacy score for iOS apps.

4.2 iShield: Privacy Score Model

In past various privacy preserving techniques have been proposed to preserve user's privacy in various domains like cloud computing, e-Health, wireless sensor networks, location based services etc. [19]. Privacy depends on sensitivity and visibility of user attributes [13]. The research study aims to preserve privacy of mobile users who use mobile apps and allow the application to access their data by granting permissions. Our previous works on dynamic analysis of run-time behavior of some iOS apps have revealed startling facts about how user's data is shared over the network [6, 25]. In this work, we extend and enhance our previous work on dynamic analysis [6, 24] and propose privacy preserving approach to decrease privacy disclosure score of an individual user. To determine the data types of the attributes that are shared over the network, an exhaustive dynamic analysis was performed for different permissions to list all types of data being shared after user grants permissions to the application. This data included details of user such as email id, password, contact number, country, region, time zone etc. as well as data related to device such as version of operating system, device model number, and memory active etc. along with permission related data. For applying privacy preserving techniques, attributes being shared across the network were categorized as (i) attributes related to user permissions, (ii) attributes related to user's personally identifiable information and (iii) attributes related to user's device.

To compute the privacy disclosure score of users, we calculate sensitivity and visibility of iOS users. For calculating the sensitivity, we consider that some attributes like phone number, medical disease, passwords, contacts, personal images and videos are more sensitive than other attributes like date of birth, country or region, time zone etc. All these factors are considered in computing sensitivity. To compute visibility of an attribute we refer the work proposed by [13]. Visibility is computed from three factors (i) accessibility to information, (ii) difficulty in extracting data and (iii) reliability of data. The overall privacy disclosure score is computed by combining sensitivity and visibility scores.

4.2.1 Calculation of sensitivity

Sensitivity computes the risk associated with attribute of a user. When sensitivity for an attribute increases, the risk posed by information disclosure of individuals users also increases. Aghasian et al. [13] used the calculated values of sensitivity by referring the work proposed by Shrivastav [26] for 11 attributes. In our work, we consider a total of 52 attributes and assign sensitivity values as derived by [13, 16] and from our previous work on dynamic analysis [24]. Sensitivity values in [16] were computed for countries US and EU, for our research study, we have selected the maximum risk value (amongst both the countries) for each attribute as its sensitivity score. These sensitivity scores corresponding to an attribute have been detailed in Table 3.

4.2.2 Calculation of visibility

Visibility determines how widely the attributes are accessible over the network. Three factors that strongly impact visibility of user's information were considered, namely (i) ease of accessibility of an attribute (ii) difficulty in extraction of data and (iii) frequency of occurrence of information disclosure i.e. data reliability. The visibility of information in these aspects will make user aware of how apps share their data over the network.

4.2.2.1 *Calculation of accessibility*

Accessibility indicates how many people have access to piece of information and to what level. For this work, we define three levels of user's information accessibility: The information with respect an attribute can be (i) accessible by

Algorithm 1: Compute Accessibility of an attribute

Compute Accessibility (Attribute: A)

Step 1: Identify the number of people who can have access to information associated with the attribute for which accessibility score is to be calculated.

Step 2: Assign L: Levels of accessibility

$L \leftarrow 1$ if the information associated with the attribute is accessible to its owner only

$L \leftarrow 2.5$ if the information associated with the attribute is accessible to friends/family/colleagues/other known people of its owner

$L \leftarrow 5$ if the information associated with the attribute is publicly accessible to everyone

Step 3: Assign accessibility score for an attribute between 1–5 based on the level of accessibility (L)

$$F_{Acc} = L, L \in \{1, 2.5, 5\}$$

return F_{Acc} ;

Table 3 Sensitivity scores for attributes

S. No.	Attributes	Sensitivity
1	Credit card number	.977
2	Bank information	.941
3	Disease status	.912
4	Passwords	.933
5	Camera live imagery	.911
6	Email content	.922
7	Microphone: live audio	.899
8	Disease symptoms	.888
9	SMS or MMS content	.867
10	Recorded video or photos	.898
11	Recorded audio	.842
12	Transaction data	.811
13	Address	.85
14	Access to other accounts	.795
15	Phone and text logs	.811
16	Health (heart rate, sleep patterns, stress, diet)	.842
17	Location tracking	.747
18	Contacts' personal information	.681
19	Political views	.683
20	Remote number connected by call	.658
21	Calendar events information	.671
22	Location tagging on pictures	.565
23	Browser history	.535
24	Current location	.501
25	Frequency of contact	.522
26	Social networking connections	.556
27	Religious views	.566
28	Device details like ID, Operating system	.511
29	Names of other Wi-Fi connected devices	.59
30	Other running apps	.424
31	Connected devices	.443
32	Relationship status	.416
33	Frequency/duration of exercise	.392
34	Bookmarked pages	.397
35	Education data	.353
36	Wi-Fi connection status	.32
37	Interests	.3
38	Internet connectivity	.293
39	Call duration	.222
40	Job details	.2
41	E-mail address	.183
42	Birth date	.116
43	Hometown	.15

(*Continued*)

Table 3 Continued

S. No.	Attributes	Sensitivity
44	Current town	.116
45	Country	.113
46	Linking accounts across platforms	.678
47	Cross- application advertising	.671
48	Predictive analysis for marketing	.619
49	Condition-targeting for advertising	.626
50	Serving third-party advertising	.593
51	Medical diagnosis	.723
52	Medication compliance	.898

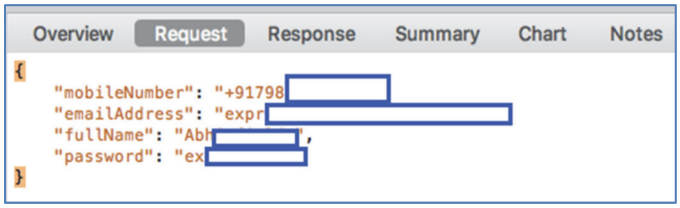

Figure 2 Data shared at the time of user's registration.

the owner of the information, (ii) accessible by friends/family/colleagues/other known people, (iii) accessible publicly to anyone. An accessibility value in between 1–5 is assigned to each attribute that is shared over the network. Algorithm 1 defines the proposed approach to compute accessibility of an attribute for a given application.

To illustrate, the assignment of level of accessibility to an attribute, we present an example of 'Doc Talk' app. The app allows patient to register and connect with doctors and share their reports. While running the application several user and device attributes were captured. Figure 2 presents a snapshot for the same application while performing user registration. Likewise, information for other attributes like medical symptoms and other personal details were also recorded and captured.

Referring Figure 2, level of accessibility for attributes mobile number, email address; full name and password are 2.5, 2.5, 2.5 and 1 respectively. As mobile number, email address and full name of a person can be known to family, friends, colleagues or other known people of the owner, whereas password should be known only to user and server. Likewise, other attributes were assigned level of accessibility. The total accessibility score $\mathrm{F}_{\mathrm{Acc}}$ for an Application A is the summation of accessibility score corresponding to all attributes involved in the communication.

4.2.2.2 *Difficulty in data extraction*

For computing privacy disclosure risk another factor that is important is difficulty in obtaining private or sensitive information from different data formats. To compute data extraction difficulty for an attribute, we define four levels of difficulty namely: low, medium, high and no extraction. A difficulty value in between 0–5 is assigned for each attribute that is shared over the network. Algorithm 2 defines the proposed approach to compute difficulty in data extraction of an attribute for a given application.

Algorithm 2: Compute difficulty in data extraction of an attribute

Compute_Difficulty (Attribute: A)

Step 1: Identify the number of people who have access to information associated with the attribute for which difficulty in data extraction has to be calculated.

Step 2: Assign L: Level of data extraction

$\mathrm{L} \leftarrow 0$ if the information associated with the attribute cannot be extracted

$\mathrm{L} \leftarrow 1$ if the information associated with the attribute can be extracted with high difficulty

$\mathrm{L} \leftarrow 2.5$ if the information associated with the attribute can be extracted with medium difficulty

$\mathrm{L} \leftarrow 5$ if the information associated with the attribute can be extracted with low difficulty

Step 3: Assign extraction score for an attribute between 0–5 based on the level of data extraction (L)

$$\mathrm{F_{Diff}} = \mathrm{L}, \mathrm{L} \in \{0, 1, 2.5, 5\}$$

return $\mathrm{F_{Diff}}$;

If the information associated with attribute is completely encrypted, and cannot be extracted the difficulty level is 0, if the information is encrypted but can be extracted with high difficulty then difficulty level is 1; if information is partially encrypted can be constructed with medium difficulty (level is 2.5) and if the information is shared in clear then the data associated with the attribute can be traced without any difficulty (level is 5).

To illustrate, the assignment of level of difficulty to an attribute, we refer to same example 'Doc Talk' app (Figure 2). Referring Figure 2, level of difficulty in data extraction for all the attributes mobile number, email address, full name and password is equal to 5. Because, we were able to trace all the attributes with low difficulty as the information associated with these attributes were shared over the network without using encryption. Likewise, other attributes were similarly assigned level of difficulty. The total difficulty score $\mathrm{F_{Diff}}$ for an Application A is calculated as summation of difficulty score of all attributes involved in the communication.

4.2.2.3 *Data reliability calculation*

Russell [27] define reliability as the extent to which one can rely on source of the data and thus, the data itself. In this context source refers to the developers of the iOS app and data relates to user attributes shared by the iOS app. Reliability computes the confidence with which a particular attribute is disclosed to one or else multiple sources [13]. Reliability of attribute will increase with the number of sources to which the information is shared. Therefore, a less value of reliability is better. We consider reliability of data disclosure for every attribute (whether user data /device data) for computing reliability. To compute the data reliability $\mathrm{F_{Rel}}$, we use Equation 1.

$$\mathrm{F_{Rel}} = \frac{2}{1 + \mathrm{e}^{-\mathrm{s}}} - 1 \tag{1}$$

Where 's' indicates number of sources to which the attribute of user/device has been revealed. In the work proposed by [13], the authors have considered 's' as number of sources across multiple online social networks. In our work, 's' includes the sources where the information of the attribute is disclosed which includes third-party domains, crash reporting frameworks, adversaries, in-app purchase, payment service partners etc. The output boundary of the function is [0, 1] in which if the number of sources to which information has been disclosed increases, reliability increases. As an example, we compute reliability of "Fitness Pal" application, which allows the users to monitor their weight, diet, calories intake etc. Figure 3 depicts the snapshot of overview, request and response of application showing sharing of device details such as version of operating system, device name, city etc. to ads.mopub.com.

Apart from this, this information was also shared to init.itunes.apple.com, p19-calendars.icloud.com, config.inmobi.com, z.moatads.com, assets.pinterest.comand a total of 81 sources.

4.2.2.4 *Total visibility calculation*

The total visibility for an attribute can be calculated using Equation 2 that sums $\mathrm{F_{Acc}}, \mathrm{F_{Diff}}$ and $\mathrm{F_{Rel}}$.

$$\mathrm{F_{vis}(x)} = \mathrm{F_{Acc}} + \mathrm{F_{Diff}} + \mathrm{F_{Rel}} \tag{2}$$

For example, in order to calculate the visibility score for attribute 'password' from 'Doc Talk' app, the calculated values for $\mathrm{F_{Acc}}, \mathrm{F_{Diff}}$ and $\mathrm{F_{Rel}}$ were 1, 5 and 0.462 respectively. Thus, the total visibility score of the attribute 'password' was computed using Equation 2 which was 6.462.

$$\mathrm{F_{vis}(password)} = 1 + 5 + 0.462 = 6.462 \tag{3}$$

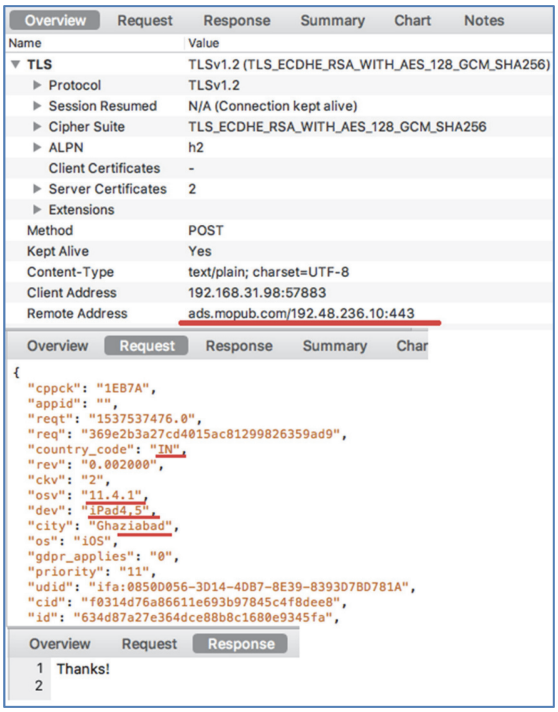

Figure 3 Details of device shared with third party advertising enterprises.

The total visibility score for an application can be calculated by summing the visibility scores for all the attributes that are shared by the application. As the visibility score of an attribute increases, the privacy disclosure risks of the individual user also increase.

4.2.2.5 *Calculation of privacy score*

As discussed in this section, privacy depends on sensitivity and visibility of an attribute. Let β_i represent sensitivity of each attribute and $F_{vis}(i)$ denote the visibility of every attribute. To compute the privacy score of attribute 'password' from 'Doc Talk' application, sensitivity score for attribute password would by multiplied with its visibility score. Thus, the privacy disclosure score for attribute password can be computed using Equation 4 and using the sensitivity score (Table 3) and visibility score (6.462).

$$\text{Privacy}_{\text{Password}} = 0.933 * 6.462 = 6.023 \tag{4}$$

The overall Privacy Disclosure Score (PDS) for a user 'u' can be calculated using Equation 5.

$$\mathrm{PDS_u} = \frac{\sum_{\mathrm{i=1}}^{\mathrm{m}} \beta_{\mathrm{i}} * \mathrm{F_{vis}(i)}}{\mathrm{m}} \tag{5}$$

Where indicates the attribute user and 'm' represents total number of attributes for an application A. As the value of privacy disclosure score increases, the user is more likely in a risk of privacy and information, whereas a less value privacy disclosure is better.

The aim of our proposed work is to reduce the privacy disclosure score. In the following section we present a case study of a health care application and calculate the privacy disclosure risk of an individual. Using dynamic analysis, we monitor the run-time behaviour of the application and capture the traffic of the application over the network using network penetration. We examine the application for 20–30 minutes which is necessary to establish proper communication between the application and its server. During testing the application, we have tried to use all the features of the application such as sign-up, login with social networking sites, adding events, mailing etc.

$\mathrm{F_{Acc}}, \mathrm{F_{Diff}}$ and $\mathrm{F_{Rel}}$ were computed for each attribute and overall privacy disclosure score was calculated using Equation 5.

4.3 Calculation of Privacy Disclosure Score of an Individual: A Case Study on Diabetes Monitoring Application

We downloaded diabetes monitoring application 'Diabetes Health Manager' from Apple's official App store. The application enables its users to monitor and store relevant health information between their clinical visits. By using the app, the users can manage their medications, treatments, track diabetes symptoms and side-effects. Users can also share their information with health care providers, access patient educating materials. Users can also gain insights from charts that capture test results and medication adherence etc. The app was run for 30 minutes to use all the features. For calculating the sensitivity of the attributes associated with the application Table 3 is been referred. During dynamic analysis we have traced all the attributes whose information has been shared over the network and tabulate them in Table 4. Table 4 depicts the attributes with respect to 'Diabetes Health Manager' application along with their sensitivity and visibility score calculated as per method described in previous section. Table 4 also depicts the computed privacy disclosure for each attribute of the application.

Table 4 Privacy disclosure score for health care application

S. No.	Attributes	S	Visibility			PDS
			F_{Acc}	F_{Diff}	F_{Rel}	
1	Disease status	.912	2.5	5	.462	7.262
2	Passwords	.933	1	5	.462	6.03
3	Camera live imagery	.911	1	0	.462	2.243
4	Disease symptoms	.888	2.5	5	.462	7.071
5	Phone and text logs	.811	2.5	5	.462	6.458
6	Transaction data/recording events	.811	2.5	5	.462	6.458
7	Health (heart rate, sleep patterns, stress, diet)	.842	2.5	5	.462	6.705
8	Contacts' personal information	.681	1	5	.462	4.401
9	Calendar information	.671	1	5	.462	4.337
10	Current location	.501	2.5	5	.462	3.989
11	Device details	.511	1	5	.999	3.577
12	Duration of exercise	.392	2.5	5	.462	3.122
13	Education data	.353	1	5	.462	2.282
14	Wi-Fi connection status	.32	1	5	.462	2.068
15	Internet connectivity	.293	1	5	.462	1.894
16	E-mail address	.183	2.5	5	.462	1.458
17	Birth date	.116	2.5	5	.462	0.924
18	Country	.113	2.5	5	.462	0.9
19	Cross- application advertising	.671	5	5	.999	7.381
20	Predictive analysis for marketing	.619	5	5	.999	6.809
21	Condition-targeting for advertising	.626	5	5	.999	6.886
22	Serving third-party advertising	.593	5	5	.999	6.523
23	Medical diagnosis	.723	2.5	5	.462	5.757
24	Medication compliance	.898	2.5	5	.462	7.15

The privacy disclosure score of a user of this app will be the privacy disclosure of all attributes (111.685) divided by total number of attributes. Referring Table 4, the number of attributes m is 24. Thus, privacy disclosure score of the individual is 4.653 which is calculated using Equation 4.

The aim of our proposed work is to decrease the privacy disclosure score of an attribute and reduce the overall privacy disclosure score of the application. As the values for sensitivity, data accessibility and data reliability cannot be changed, therefore we propose certain ways in which the difficulty level in data extraction can be increased. Following section proposes ways to reduce privacy disclosure score of attributes.

5 iShield Framework: Privacy Preserving Methods

In this work, we propose privacy preserving methods that can be adopted by the developers of the application before sharing user's data.

5.1 Data Type Specific Privacy Preserving Approach

Data type of the attributes shared by mobile apps has different value ranges. For example, when a user allows an application to access his health-kit, the type of variables associated with health-kit permission can include height, body mass, body mass index, blood glucose etc. Depending upon the functionality of the applications the parameters can vary. After identifying the variables specific to permissions, their data-types were identified and were mapped to standard data types used in data preserving approaches.

Data Objects and Attribute Types: Data objects are the mobile applications and the attributes shared by the applications. Type of an attribute is determined by a set of possible values such as nominal, binary, numeric or ordinal which are considered as basic data types [28]. Apart from these, the application can also share other attribute like strings, time, phone number etc. In this work these types have been handled separately and defined in the subsequent section.

5.1.1 Nominal or categorical attribute

The term nominal is related to names. The possible values of nominal attribute are symbols or names of things. The values can represent some sort of category, code etc. For example, the location permission takes country code as a parameter. The possible values for country code are AL, AU, BH, BD, BR, CN, IN, ID etc.

5.1.2 Binary or Boolean attribute

A binary attribute is a nominal attribute with only two states 0 or 1, where 0 may imply that the attribute is not present and 1 implies that the attribute is present. Binary attributes that have two states true or false are referred to as Boolean [18]. For example, in case of camera permission, whether the device supports video orientation or not is a binary variable because it can have either true or false as its value.

5.1.3 Ordinal attribute

An ordinal attribute is an attribute with possible values that have an order or ranking amongst them. For example, in case of health-kit permission, blood

type parameter can have values such as AB+, AB−, A+, A−, B+, B−, O+ and O−.

5.1.4 Numeric attribute

A numeric attribute is a measurable quantity, which is represented by integers or real values. The attributes can be interval-scaled or ratio-scaled. The interval-scaled attributes are measured on scale of equal-size units. For example, in health-kit permission attribute body-temperature is interval scaled [18]. Other examples can include calendar dates in case of calendar permission. The ratio-scaled attribute has all attributes of interval-scaled variables and one additional attribute inherent "zero-point". For example, in case of health-kit permission, attribute blood glucose (measured in mmol/L or mg/dL) represents a ratio-scale.

5.1.5 Strings, date and time zone

Apart from the above standard attributes, dynamic analysis studies have identified that there are a lot of string variables that shared over the network. Examples include user name, password, email, tweets, state, country etc. Additionally, dates and time zone are also shared. Therefore, we have also suggested technique to preserve strings, date and time zone which is easy for developers to integrate in application's code as well as cost effective.

5.1.6 Phone number, zip-code

During dynamic analysis, it was identified that most of the applications require user's mobile number to register users, send one-time password (OTP) or SMS and were not encrypted for most of the apps. Many apps seeking location permission had sent zip-codes without encryption. Therefore, we propose encoding technique that can be adopted to preserve phone number and zip-code.

5.1.7 Multimedia files: image, audio, video

Other types of data that are most commonly shared over the network include multimedia files such as image, audio and video. During dynamic analysis it was identified that most of the iOS apps share images, video and audio files but never protect the contents. The prime reason is that the URL's through which the data is shared remains active on the hosts and is easily accessible [5, 24]. Standard multimedia encryption techniques can be used to protect multimedia contents stored in App's cloud. These techniques have not been described in this paper.

5.2 Application of Privacy Preserving Techniques to Protect user's Privacy

Many researchers have focused their studies on a single type of permission such as location [1, 20], health [29], instant messages [30] and how to preserve data associated with these permissions. In this paper, we have applied several data type specific data perturbation techniques to preserve user's privacy [19]. All the techniques can be applied to live data that was shared while the application was running in foreground. The techniques have been demonstrated with the (i) help of snapshots that were captured through network penetration, (ii) numeric calculations (for numeric attributes) at sender/receiver, (iii) strategies for encoding data (for Boolean and categorical attributes) and (iv) data reconstruction at server side. The proposed data perturbation techniques are cost effective than classical encryption techniques.

5.2.1 Classification of privacy preserving techniques

Existing privacy preserving techniques are divided into three categories: data perturbation, data encryption and data anonymization techniques.

5.2.1.1 *Data perturbation*

Data perturbation techniques replace the sensitive information of original data set by applying series of operations [17]. Examples include replacing the sensitive information through anonymous perturbation, appending random variables, replacement, adding perturbation information to release and compute. The main advantage of using data perturbation of technique is that it has less computing cost and is easy to implement.

5.2.1.2 *Data encryption*

The method uses encryption techniques to hide sensitive data. Data encryption is widely used in distributed application environment and guarantees authenticity, non-destructiveness of data, reversibility and thereby increasing degree of privacy preserving [17]. Major disadvantage is computation overhead, communication overhead and deployment complexity. We propose use of data perturbation techniques, wherever possible, in place of data encryption techniques because the perturbation techniques are easy to implement and have less computing and communication overhead.

Table 5 Data types and their suggested data perturbation techniques

Data Type	Data Perturbation Techniques
Numeric	Min-Max normalization technique
Categorical	Numeric encoding/decoding technique
Boolean	Character randomization technique
Ordinal	Numeric encoding/decoding technique
String	Conversion into hexadecimal ASCII with Auto key encryption
Date & Time	Conversion into hexadecimal ASCII with Auto key
Phone number, Zip-code	Character randomization technique

5.2.1.3 *Data anonymization*

The technique hides the identity of the user and all sensitive data to preserve privacy [17]. This technique can be applied only when both sender and receiver have already shared the data securely and anonymized fields are sent just for sake of partial authentication. For example, only last 2 digits of a phone number can be sent and remaining can be masked. This method can't be applied in our problem, as all data is fresh and app server needs to reconstruct it on reception.

Table 5 enumerates the data type specific perturbation techniques applied to reduce the visibility of attributes according to their data type. All the techniques have been demonstrated for the attributes that were captured during dynamic analysis of 'Diabetes Health Manager' application.

5.2.2 Data type specific privacy preserving techniques

5.2.2.1 *Numeric data type*

For reducing the visibility of numeric attributes, we consider the 'Diabetes Health Manager'. Several functionalities of the application were tested like user registration, sign-up, recording transactions etc. Figure 4 depicts snapshot of an activity that was captured when performing user registration with details of the attributes that were shared. Referring Figure 4, we can depict that user's personal attributes like date of birth, cholesterol, HbA1c details, current weight, height, target weight, zip code, state; email address are being shared in clear text. In order to secure numeric values of user's data, we propose data perturbation techniques using max-min normalization.

The values are normalized within an assumed range [21]. The range can be selected by the developers and would be known to client and server for perturbation and data reconstruction respectively. To map a 'v' value, for an attribute 'A' within a specified range $[\min_A, \max_A]$ to a new range

Figure 4 Shared attributes of user during user registration.

$[\text{new\_min}_\text{A}, \text{new\_max}_\text{A}]$ is given by the Equation 6.

$$\text{P} = \frac{\text{v} - \text{min}_\text{A}}{\text{max}_\text{A} - \text{min}_\text{A}}(\text{new\_max}_\text{A} - \text{new\_min}_\text{A}) + \text{new\_min}_\text{A} \qquad (6)$$

Where P: is the perturbed value for the attribute in the required range. Values for $\text{min}_\text{A} = 1, \text{max}_\text{A} = 1000, \text{new\_min}_\text{A} = 8$ and $\text{new\_max}_\text{A} = 2000$ were used for perturbing attributes. Application developers can assume common values for mapping all attributes or they can select different ranges for different attributes. For our computation we have assumed common values for perturbing all attributes. Table 6 enumerates perturbed values of chosen app after applying Min-Max normalization technique.

From Table 6 we can infer that the data of attributes age weight, blood sugar reading have completely changed thereby preserving user's privacy. Now this data can be sent from the app over the network.

User attributes can be reconstructed at receivers' side by using Equation 7 [21].

$$\text{v} = \frac{\text{P} - \text{new\_min}_\text{A}}{\text{new\_max}_\text{A} - \text{new\_min}_\text{A}}(\text{max}_\text{A} - \text{min}_\text{A}) + \text{min}_\text{A} \qquad (7)$$

Table 6 Perturbed values of numeric attributes after min-max normalization at sender's side

Attributes	Original	$\min_A$	$\max_A$	new_$\min_A$	new_$\max_A$	Perturbed
Age	33	1	1000	8	2000	71.8078078
Height	180	1	1000	8	2000	364.924924
Current weight	85	1	1000	8	2000	175.495495
Target Weight	75	1	1000	8	2000	155.555555
Cholesterol	129	1	1000	8	2000	263.231231
HbA1c	8.5	1	1000	8	2000	22.9549549
gulcose_before	88	1	1000	8	2000	181.477477
gulcose_after	140	1	1000	8	2000	285.165165
insulin-unit	1	1	1000	8	2000	8
carbohydrate	500	1	1000	8	2000	995.003

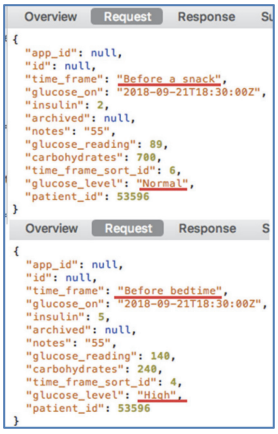

Figure 5 Details of medical events shared via application.

Data perturbation techniques can reduce the visibility level of attributes and reduce the privacy disclosure score of individuals. The technique is cost effective and can be easily adopted by developers of the application to protect user's privacy.

5.2.2.2 *Categorical data type*

For perturbing attributes having categorical values, we propose that the developers of the application can use numeric encoding techniques to encode the attributes. For example, while testing the application 'Diabetes Health Manager' app, we added certain events to add blood sugar levels at different time periods. Figure 5 depicts the snapshot that has combined details for two recorded events. Referring Figure 5, we can depict that attributes time_frame

Table 7 Data perturbation technique for categorical data type

Attribute	Categorical Values	Numeric Encoding
Time frame	Before breakfast	1
	After breakfast	2
	Before lunch	3
	After lunch	4
	Before bedtime	5
	During the night	6
	Before a snack	7
	After a snack	8
Glucose level	High	1
	Normal	2
	Low	3
	Severe low	4

and glucose_level belong to categorical data type. Therefore, to reduce the privacy disclosure score for these attributes, numeric encoding/decoding techniques can be used at senders and receivers' side. The motivation behind numeric encoding is that string data-type is easy to trace and understand instead of numeric data. From Figure 5 one can easily figure out that the person is having normal glucose level before a snack. Such crucial information of the user must be preserved. Thus, if the developers use numeric code '2' to transmit 'normal glucose level' the visibility for the attribute will be reduced as it would be difficult for the attackers to decode all such values. Similarly, other categorical attributes can be encoded. Table 7 depicts the technique for encoding categorical data.

At the receiver's end numeric decoding can be applied to recover the original values of attributes.

5.2.2.3 *Boolean data type*

Another type of data that is shared across the network is Boolean. From Figure 4, we can see that there are lots of attributes which are Boolean valued. For example, the attributes allergies_set, food_allergies, last_login_alert, medication_allergies etc. are Boolean valued. To perturb Boolean data types we propose Algorithms 3, 4 which can be applied by the developers.

Sending data from sender

Table 8 enumerates the original and perturbed values after applying the data perturbation technique proposed at senders' side. By using Table 8, we can perturb the original values of the Boolean-valued attributes.

Algorithm 3: Encoding Boolean attribute at sender's side

Encode_ value (Boolean: B)
Step 1: If value to be assigned to a Boolean attribute B is TRUE then
Step 2: Generate a random character between [a-m] and assign it to the Boolean attribute.
Step 3: Else if value to be assigned to a Boolean attribute B is FALSE then
Step 4: Generate a random character between [n-z] and assign it to the Boolean attribute
return character;

Algorithm 4: Decoding at receiver's side to reconstruct original value of Boolean attribute

Decode_ value (Character: C)

Step 1: If the received character belongs in-between [a-m]: assign value TRUE or 1 to the Boolean attribute

Step 2: Else if the received character belongs in-between [n-z]: assign value FALSE or 0 to the attribute

return Boolean;

Table 8 Data perturbation technique for Boolean data type

Attribute	Actual Value	Perturbed Value using Random Character Generation
allergies_set	False	o
food_allergies	False	q
last_login_alert	True	c
medication_allergies	False	t
reset_password	False	u
treatment_schedule_alert	True	f

Data reconstruction at receiver's side

For reconstructing data at the receiver rules defined in table are used. Table 9 depicts the actual values of the attributes after reconstruction at receiver's end.

5.2.2.4 *Ordinal data type*

Ordinal values are typically used to measure non-numeric concepts with scale like satisfaction, happiness etc. For encoding the ordinal values numeric encoding can be used in place of string. Table 10 depicts the proposed technique to encode ordinal data for an attribute that was captured from

Table 9 Reconstructed values at receiver's side

Attribute	Perturbed Value using Random Character Generation	Actual Reconstructed Value
allergies_set	o	False
food_allergies	q	False
last_login_alert	c	True
medication_allergies	t	False
reset_password	u	False
treatment_schedule_alert	f	True

Table 10 Data perturbation technique for ordinal data type

Attribute	Ordinal Values	Numeric Encoding
My mood today?	Angry	1
	Anxious	2
	Calm	3
	Content	4
	Depressed	5
	Edgy	6
	Happy	7
	Irritated	8
	Nervous	9
	Sad	10
	Sleepy	11
	Stressed	12
	Tired	13
	Worried	14

'Diabetes Health Manager' app. The numeric encoded data can be sent from sender. While at receiver the reverse method can be applied to reconstruct the original data.

5.2.2.5 *Strings, date and time zone data type*

Strings: Another type of data which is commonly shared over the network is strings. Examples include name, address, tweets, instant messages etc. Figure 6 represents a snap shot for the application when user performs a login.

From Figure 6 we can easily see that user's personal details such as password and email address are shared in clear text and can be accessed very easily. In order to preserve strings and their data, auto key generation technique can be used for enhancing the privacy level. Sender and receiver select a small key. The remaining key is auto generated from the text to be encrypted. Encryption is based upon a XOR operation between key and plaintext data.

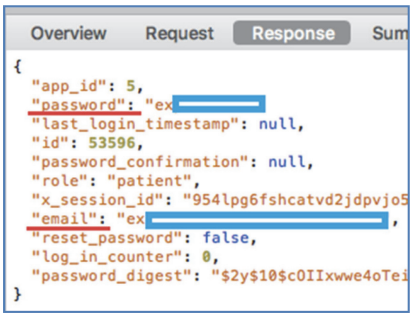

Figure 6 User's shared attributes at the time of login.

At the receiver's side, as the receiver already knows the key, it performs a XOR operation of key with the string sent by the sender to get actual string. Algorithms 5 and 6 depict the proposed technique of encoding and decoding strings for sender and receiver respectively.

The method proposed in Algorithms 5 and 6 are demonstrated with the help of calculations for the data which is shared by the application.

Algorithm 5: Encoding String at Sender

Encode_String (String: S, Auto-key: A)
Declarations
LENGTH// length for character array
plaintext[LENGTH]//stores the plaintext, here S is plaintext
key[LENGTH]//stores the key
XOR[LENGTH]//stores the cipher text
Integer i, len1, len2
Step 1: len1 = strlen(plaintext)
Step 2: len2 = strlen(key)
Step 3:
for i = len2 to i $\leq$ len1 Step 4
Step 4: key[i] = plaintext[i = len2]
Step 5: Increment the counter, i
End for
Step 6: key[i] = $'\backslash 0'$
Step 7:
for i = 0 to i $\leq$ len1 Step 8
Step 8: XOR[i] = (char)plaintext[i] $\oplus$ key[i]
Step 9: Increment the counter, i
Return XOR[i] //returns the ciphertext

Algorithm 6: Decoding String at Receiver

Decode_String (Encoded_String: XOR [i], Auto-key: A)
Declarations
LENGTH// length for character array
ciphertext[LENGTH]//stores the ciphertext
rkey[LENGTH]//stores the key
Integer i, l1, l2, m, k, k1 = 0
Step 1: len1 = strlen(ciphertext)
Step 2:
for i = 0 to i $\leq$ len1 Step 3 //len1 denotes the length of ciphertext
Step 3: ciphertext [i] = XOR[i]
End for
Step 4: ciphertext [i] = $'\backslash 0'$ //appends null character
Step 5: rkey[LENGTH]//stores the key
Step 6: l2 = strlen(rkey)
Step 7: m = len1/l2
Step 8:
outer for k = 0 to k $\leq$ m Step 9
Step 9:
inner for i = 0 to i < l2 Step 10
Step 10:
XOR[i + (k) $*$ (l2)] = (char)(ciphertext[i + (k) $*$ (l2)])$\oplus$rkey[i])
Step 11:
rkey[i] = XOR[i + (k) $*$ (l2)]
Step 12: Increment the counter, i
end inner for
Step 13: rkey[i] = $'\backslash 0'$
Step 14: k1 = i + (k) $*$ (l2)
Step 15: Increment the counter, k
end outer for
Step 16:
for i = 0 to i < l1%l2 Step 17
Step 17: XOR[k1] = (char)(ciphertext[k1] $\hat{}$ rkey[i]);
Step 18: Increment the counter, k1 and end for
Step 19: for i = 0 to i < l1
Step 20: Return XOR[k1] and end for // **returns the plaintext**

Sending data from sender: illustration with example

Referring Figure 6, suppose user enters email as 'expressway@gmail.com' then the privacy of this attribute can be enhanced by encoding its value as proposed in Algorithm 5.

Step 1: Store the plaintext in a matrix and calculate its length.

Plaintext: 'expressway@gmail.com'

Table 11 Generation of auto key according to length of plaintext

S	e	x	p	r	e	s	s	w	a	y	@	g	m	a	i	l	.	c	o	m
K	d	e	f	e	n	d	e	x	p	r	e	s	s	w	a	y	@	g	m	a

Step 2: Both sender and receiver select a common key. Suppose that the key is 'defend'. Store the key in a matrix and calculate its length.

Key: 'defend'

Step 3: Generate key stream by placing the key before the plaintext and then placing the characters from plaintext till the length of the plain text. Table 11 describes the plain text and key.

Step 4: Perform a XOR operation of the plaintext and key to generate cipher text. The string the can be represented using standard representations like binary, hexadecimal etc. In this context, we have used hexadecimal notation.
Ciphertext: 11d1617b1716f11b25141e168156e42c
Similarly, all the other strings can be encoded and then transmitted over the network.

Data reconstruction at receiver's side: illustration with example

To reconstruct the data the receiver's side Algorithm 6 can be applied.

Step 1: Receive the ciphertext

Ciphertext: 11d1617b1716f11b25141e168156e42c

Step 2: As the receiver already knows the key, it performs an XOR operation with the ciphertext to reconstruct the plaintext using a common key. The XOR operation generates characters which are equal to the length of the key. These characters would be again XORED with the remaining characters of the ciphertext to generate key. The process continues till the length of ciphertext.

Plaintext: expressway@gmail.com
To perturb date and time zone, encoding techniques proposed in Algorithms 5 and 6 can be applied.

5.2.2.6 *Phone number and zip-code*

Another type of user's data that is commonly shared over the network is phone number and zip-code (once the user grants location permission). During dynamic analysis of applications, it was observed that user's phone number and zip-codes are not encrypted resulting into privacy breach of users (shown in Figures 2, 4 and 7). To perturb this data, Algorithms 7 and 8 are proposed that can be applied at sender and receiver respectively.

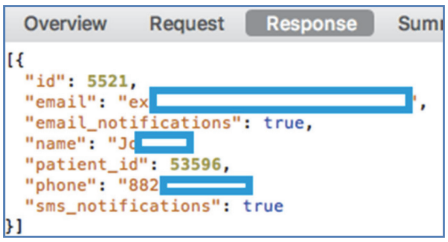

Figure 7 User's personal details shared over network.

Algorithm 7: Encoding phone number at sender's side

Encode_number (Number: N)
Step 1: If N = 9, Generate a random character between [a-c] and assign the character to N
Step 2: Else if N = 8, Generate a random character between [d-f] and assign the character to N
Step 3: Else if N = 7, Generate a random character between [g-h] and assign the character to N
Step 4: Else if N = 6, Generate a random character between [i-j] and assign the character to N
Step 5: Else if N = 5, Generate a random character between [k-m] and assign the character to N
Step 6: Else if N = 4, Generate a random character between [n-p] and assign the character to N
Step 7: Else if N = 3, Generate a random character between [q-s] and assign the character to N
Step 8: Else if N = 2, Generate a random character between [t-v] and assign the character to N
Step 9: Else if N = 1, Generate a random character between [w-x] and assign the character to N
Step 10: Else if N = 0, Generate a random character between [y-z] and assign the character to N
return character;

Data reconstruction at receiver's side

To reconstruct the data the receiver's side Algorithm 8 can be applied.

Illustration with example

At sender's side

Phone number = 8738357800
After applying data perturbation using Algorithm 8
Resultant string: dhqeskhfzy

Algorithm 8: Reconstructing phone number at receiver's side

Decode_character (Character: C)
Step 1: If the received character belongs in-between [a-c] then N = 9
Step 2: Else if the received character belongs in-between [d-f] then N = 8
Step 3: Else if the received character belongs in-between [g-h] then N = 7
Step 4: Else if the received character belongs in-between [i-j] then N = 6
Step 5: Else if the received character belongs in-between [k-m] then N = 5
Step 6: Else if the received character belongs in-between [n-p] then N = 4
Step 7: Else if the received character belongs in-between [q-s] then N = 3
Step 8: Else if the received character belongs in-between [t-v] then N = 2
Step 9: Else if the received character belongs in-between [w-x] then N = 1
Step 10: Else if the received character belongs in-between [y-z] then N = 0
return phone number;

At receiver's side
Received string:dhqeskhfzy
Phone number = 8738357800

The advantage of using the proposed technique is that every time a different character would be assigned to same number which is difficult to decode.

With these techniques in place, user's data associated of all types of attributes is protected, thereby increasing the privacy level of individual users.

6 iShield Framework: Performance Evaluation

The section describes experimental evaluation of iShield framework and its impact on privacy disclosure on an individual user. We also compared iShield with state of the art approach.

6.1 Reduction in Privacy Disclosure Score

We have taken example of the 'Diabetes Health Manager' application. In addition to it we analyzed and reduced the privacy disclosure of 10 popular iOS health care apps. To demonstrate the empirical results, we calculated the privacy disclosure score for the apps before and after applying privacy preserving strategies. Table 12 depicts the results of privacy disclosure score. A detailed case study of 'Diabetes Health Manager' application is depicted in Table 13.

From Table 12, we can infer that the proposed techniques have reduced the privacy disclosure score of the applications from a min of 4.385% to a

Table 12 Results of privacy disclosure score for popular health care iOS apps

S. No.	Application	PDS_Before	PDS_After	Privacy Quantified (%)
1	My Fitness Pal	3.975	1.962	50.64
2	Medscape	2.645	2.529	4.385
3	Yazio	4.407	2.268	48.53
4	Doctalk	3.556	1.817	48.90
5	One Touch	4.099	2.037	50.30
6	Diabetes M	4.117	2.119	48.53
7	Fit Pass	3.483	1.876	46.13
8	Seven Workouts	4.308	2.163	49.80
9	Doctor Insta	4.389	2.366	46.10
10	Social Diabetes	3.88	1.636	57.84

maximum of 57.84%. The techniques proposed are applicable to all application categories as well as for all types of permission.

Privacy disclosure score for the 'Diabetes Health Manager' app was 4.653 (refer Table 4). Table 13 shows the updated values of the visibility scores corresponding to all the attributes of the 'Diabetes Health Manager' application after applying privacy preserving strategies. The privacy disclosure score of the app after applying privacy preserving technique is 1.967. From, Table 4 (Section 4) and Table 13 we can deduce that the overall privacy disclosure score of the application has been reduced by 57.72%.

6.2 Comparison of iShield with State of the Art Approach

We have compared iShield with existing work on πBox, a platform for privacy preserving of apps [2]. The platform allows the users to run the un-trusted apps on trusted platform. We compared our work with the help of privacy preserving metric privacy disclosure score of a user. Lee et al. [2] have stated that πBox sharing channel guarantees that only specified recipient can read shared content, whereas a malicious app may hide private information via steganography. In such cases πBox shows content to be shared to user and then use power-box to confirm user's content to be shared. Only plain text and images are allowed in their prototype. However, the approach lacks other data types such as numeric, nominal, binary and ordinal. Their proposed technique does not encrypt or encode users' data with different data types, thus does not guarantee to preserve end user's privacy. However, iShield supports several data formats and provides data type specific privacy preserving techniques thus guarantees to preserve all types of users' data. Lee et al. [2] highlight that there is a trade-off between usability and privacy. Therefore, πBox avoids involving

Table 13 Privacy disclosure score (PDS) after applying privacy preserving techniques

S. No.	Attributes	Actual Value being Sent		Perturbed Value	S	Visibility			PDS
		Parameter Name	Value			F_{Acc}	F_{Diff}	F_{Rel}	
1	Disease status	id	14	33.92192192	.912	2.5	1	.462	3.614
		Medical_ condition_id	14	33.92192192					
		Patient_id	53596	106876.1081					
		medical_ condition_denormalized	Asthma	3612d291e15					
2	Passwords	password	express900	12199171d31 5c4343	.933	1	1	.462	2.297
3	Camera live imagery	caption	My car	3a18592e1852	.911	1	0	.462	1.331
		size	742.86	1487.264384					
		content-type	image/jpeg	1ec18e84ed15 4ad					
4	Disease symptoms	symptom_ids	5011,5012	9997.90991, 9999.903904	.888	2.5	1	.462	3.518
5	Phone and text logs	phone	8842267907	edqvuifayf	.811	2.5	1	.462	3.213
6	Transaction data recording events	Pain_rating	3	11.98798799	.811	2.5	1	.462	3.213
		Journal_on	2018-09-19T18:30:00Z	455148a1d 1101079016 e2803060					
		weight	88	181.4774775					
		bp_diastolic	134	273.2012012					
		notes	My daily records	3a18592918 49818491e1 c431d1771c					
		bp_systolic	89	183.4714715					

(*Continued*)

Table 13 Continued

S. No.	Attributes	Actual Value being Sent		Perturbed Value	S	Visibility			PDS
		Parameter Name	Value			F_{Acc}	F_{Diff}	F_{Rel}	
7	Health (diet, sleep, heart rate)	carbohydrates	700	1401.801802	.842	2.5	1	.462	3.336
8	Contacts' personal information	phone	8826464312	deujoirwu	.681	1	1	.462	1.676
		id	5521	11014.84685					
		sms_notifications	1	8					
		email	expressgarden 1314@gmail. com	12199171d31 2121341c555 65f573565921 e14fa343					
		email_notifications	1	8					
		name	Joy	3de0					
		patient_id	53596	106876.1081					
		phone	8826464312	deujoirwu					
9	Calendar information	calendar_on	1	8	.671	1	1	.462	1.652
10	Current location	city	Ghaziabad	309183d101885	.501	2.5	1	.462	1.985
		state	Uttar Pradesh	2215d346543104 1341712c					
		Zip code	201014	vywzxo					
11	Device Details	manufacturer	Apple	361192d15	.511	1	1	.999	1.533
		model	iPad4,5	1e3118d644d51					
		version	11.4.1	56444f464a56					
12	Duration of exercise	activity	Swimming	2416103e1a03a	.392	2.5	1	.462	1.553
		duration	24	53.86186186					
		activity_intensity	Intense	3efd2c070					

Table 13 Continued

S. No.	Attributes	Actual Value being Sent		Perturbed Value	S	Visibility			PDS
		Parameter Name	Value			F_{Acc}	F_{Diff}	F_{Rel}	
13	Education data	educational_ reading_denormalized	Heart and Blood Vessel Problems	3f4183a1141131 a44412c84f2d84 f391532501f533 51e4f321eaf1f	.353	1	1	.462	0.869
		educational_ reading_id	15	35.91591592					
		educational_ reading_url	http://www.cdc. gov/diabetes/ managing/ problems.html	1f15d184e5b5f4 d58585914134d 4d3c5848b1f4e6 c157165b81241c 67f0461e154012 1eaf1f4b57434					
14	Wi-Fi connection status	wifi	true	d	.32	1	1	.462	.787
15	Internet connectivity	Network: {“cellular”, “wifi”}	false, true	x, f	.293	1	1	.462	.721
16	E-mail address	email	expressgarden 1215@gmail.com	12199171d3121 21341c55575f4 72565821e14f a343	.183	2.5	1	.462	.725
17	Birth date	dob	1994-12- 31T00:00:00Z	465840514860 2379316e000 0060	.116	2.5	1	.462	.459

(*Continued*)

Table 13 Continued

S. No.	Attributes	Actual Value being Sent		Perturbed Value	S	Visibility			PDS
		Parameter Name	Value			F_{Acc}	F_{Diff}	F_{Rel}	
18	Country	locale	En_IN	32f26c20	.113	2.5	1	.462	.447
		time zone	Asia/Kolkata	361210205c22 e4320e18a					
19	Medical diagnosis	time_frame	Before a snack	3541f2d1734f1 34553f41105	.723	2.5	1	.462	2.865
		glucose_on	2018-09-21T18:30:00Z	455148a1d1102 879396e2803060					
		insulin	2	9.993993994					
		Glucose_reading	89	183.4714715					
		Time_frame_sort_id	6	17.96996997					
		glusose_level	normal	19eb3e1e					
		Patient_id	53596	106876.1081					
20	Medication compliance	Treatment_type_id	1	8	.898	2.5	1	.462	3.558
		Treatment_name	Insulin check-up	3efa3c21a1b4c a6450348161b					
		Current_schedule_on	2018-09-25T06:03:08Z	455148a1d11 02c79236e05 00b60					
		Medication_id	5055	10085.64565					
		Treatment_duration	45	95.73573574					
		Start_on	2018-09-25T06:03:08Z	455148a1d1102 c79236e0500b60					
		End_on	2018-11-09T06:03:08Z	455148a1d09018 790f6e0500b60					
		treatment_ schedule_ denormalized	3 times per day	4441d5a4d111 a4d15165250 11359					

users in privacy critical decisions. At the same time, they highlight that sharing data is important for many apps. In such cases, πBox lets users explicitly accept a privacy risk when sharing content. However, iShield doesn't require intervention when user is running the app and has already granted permissions to the app. Instead it applies privacy preserving techniques to preserve end user's privacy when integrated and implemented during app creation. The advantage of iShield is that users need not be concerned about apps intentions, and even if the user has approved all the permissions requested by the app, the data type specific privacy preserving techniques will protect the data. Therefore, advantage of using iShield over πBox is that it eliminates the acceptances of privacy risk by the users as well as does not require user's intervention. πBox is especially suitable for enterprises where apps contain content from single publisher, do not require sharing of content, do not rely on advertisements, and do not engage in functionalities with various external parties like brokered ad-auctions etc. Whereas, iShield is free from such restrictions as it is designed to protect data associated with permissions.

7 Conclusion

With increasing popularity of Smartphones, billions of applications have been made available on online stores. As Smartphone contains a lot of user's private and confidential information, it has also gained attraction from lot of adversaries. It generates contextual data through its sensors and can constantly collect data to offer or improve its existing services, thereby raising privacy concerns. In this work, privacy preserving techniques have been proposed as part of iShield framework to assist developers in preserving privacy. Evaluation of the proposed framework by analysis of privacy disclosure score to assess the privacy level of individual was done. iShield framework enhanced privacy level of user by 57.72%.

References

[1] S. Zhong, L. Li, Y. G. Liu, and Y. R. Yang, "Privacy-preserving location-based services for mobile users in wireless networks," Yale Comput. Sci. Tech. Rep. YALEU/DCS/TR-1297, pp. 1–13, 2004.

[2] S. Lee, E. L. Wong, D. Goel, M. Dahlin, and V. Shmatikov, "πBox: a platform for privacy-preserving apps," 10th USENIX Conf. Networked Syst. Des. Implement., pp. 501–514, 2013.

[3] Rafia Shaikh, "Thousands of iOS & Android Apps Are Leaking Data of Millions of Users." [Online]. Available: https://wccftech.com/thousands-ios-android-apps-leaking-data/. [Accessed: 12-Sep-2018].
[4] Rene Millman, "Too many apps leak personal data to third parties, report finds." [Online]. Available: https://www.scmagazineuk.com/apps-leak-personal-data-third-parties-report-finds/article/1479383. [Accessed: 12-Sep-2018].
[5] D. Walnycky et al., "Network and device forensic analysis of Android social-messaging applications," vol. 14, 2015.
[6] A. J. Bhatt, C. Gupta, and S. Mittal, "Network Forensics Analysis of iOS Social Networking and Messaging Apps," in Eleventh International Conference on Contemporary Computing (IC3), Noida, India, 2018, pp. 324–329.
[7] Apple Inc., "App Sandboxing – Apple Developer." [Online]. Available: https://developer.apple.com/app-sandboxing/. [Accessed: 03-Nov-2018].
[8] M. Mohanty, "iOS SANDBOXING Lecture 16 RECAP?: DEP AND ASLR Buffer overflow attack can happen," 2018.
[9] Bart Jacobs, "What Is Application Sandboxing." [Online]. Available: https://cocoacasts.com/what-is-application-sandboxing. [Accessed: 03-Nov-2018].
[10] W. Fang, X. Z. Wen, Y. Zheng, and M. Zhou, "A Survey of Big Data Security and Privacy Preserving," IETE Tech. Rev., vol. 4602, no. September, pp. 1–17, 2016.
[11] Kees Friesland, "The Pros and Cons of Data Encryption – TechNadu." [Online]. Available: https://www.technadu.com/pros-and-cons-of-data-encryption/38599/. [Accessed: 03-Nov-2018].
[12] Steve Lander, "Disadvantages of Public Key Encryption." [Online]. Available: https://smallbusiness.chron.com/disadvantages-public-key-encryption-68149.html. [Accessed: 03-Nov-2018].
[13] E. Aghasian, S. Garg, and L. Gao, "Scoring Users Privacy Disclosure Across Multiple Online Social Networks," IEEE Access, vol. 5, pp. 13118–13130, 2017.
[14] Proofpoint Staff, "Is nothing sacred? Risky mobile apps steal data and spy on users." [Online]. Available: https://www.proofpoint.com/us/threat-insight/post/Risky-Mobile-Apps-Steal-Data. [Accessed: 12-Sep-2018].
[15] Appthority, "Mobile App Reputation Report," 2014.
[16] Kryptowire and IAPP, "Assessing Mobile App Data Privacy Risk," 2017.

[17] W. Fang, X. Z. Wen, Y. Zheng, and M. Zhou, "A Survey of Big Data Security and Privacy Preserving," IETE Tech. Rev., vol. 4602, no. September, pp. 1–17, 2016.
[18] E. Bertino, D. Lin, and W. Jiang, "A Survey of Quantification of Privacy Preserving Data Mining Algorithms," PrivacyPreserving Data Min., vol. 34, pp. 183–205, 2008.
[19] R. Mendes and J. P. Vilela, "Privacy-Preserving Data Mining: Methods, Metrics, and Applications," IEEE Access, vol. 5, pp. 10562–10582, 2017.
[20] K. Gao, Y. Zhu, S. Gong, and H. Tan, "Location privacy protection algorithm for mobile networks," Eurasip J. Wirel. Commun. Netw., vol. 2016, no. 1, 2016.
[21] C. Saranya and G. Manikandan, "A study on normalization techniques for privacy preserving data mining," Int. J. Eng. Technol., vol. 5, no. 3, pp. 2701–2704, 2013.
[22] E. Sy, T. Mueller, and D. Herrmann, "AppPETs?: A Framework for Privacy-Preserving Apps."
[23] R. Liu, J. Cao, S. VanSyckel, and W. Gao, "PriMe: Human-centric privacy measurement based on user preferences towards data sharing in mobile participatory sensing systems," 2016 IEEE Int. Conf. Pervasive Comput. Commun., pp. 1–8, 2016.
[24] A. J. Bhatt, C. Gupta, and S. Mittal, "iABC: Towards a hybrid framework for analyzing and classifying behaviour of iOS applications using static and dynamic analysis," J. Inf. Secur. Appl., vol. 41, pp. 144–158, 2018.
[25] A. J. Bhatt, C. Gupta, and S. Mittal, "iABC-AL: Active learning-based privacy leaks threat detection for iOS applications," J. King Saud Univ. – Comput. Inf. Sci., 2018.
[26] J. L. Becker and H. Chen, "Measuring privacy risk in online social networks," pp. 2095–2100, 2009.
[27] B. Russell, "Evaluating Information?: Validity, Reliability, Accuracy," Sage Res. Methods, pp. 79–99, 2007.
[28] Han Jiawei; Kamber Micheline; Pei Jian, Data Mining Concepts and Techniques, Third. 2012.
[29] H. Lee, S. Kim, J. W. Kim, and Y. D. Chung, "Utility-preserving anonymization for health data publishing," BMC Med. Inform. Decis. Mak., vol. 17, no. 1, pp. 1–12, 2017.
[30] Z. Wang, Z. Ma, S. Luo, and H. Gao, "Enhanced Instant Message Security and Privacy Protection Scheme for Mobile Social Network Systems," vol. XX, no. c, 2018.

Biographies

Arpita Jadhav Bhatt is Assistant Professor (Grade-II) in the Department of Computer Science & IT from Jaypee Institute of Information and Technology, Noida, India. She obtained her Masters in Engineering in Software Systems from Birla Institute of Technology and Science, Pilani (BITS Pilani) in 2010 and Bachelor of Technology degree from Rishiraj Institute of Technology, Indore in 2008. Her areas of interest are mobile application engineering, protecting data leaks from mobile apps, software engineering, programming in iOS, mobile computing.

Chetna Gupta is Associate Professor in the Department of Computer Science & IT from Jaypee Institute of Information and Technology, Noida, India. She obtained her Doctorate in the area of Software Testing. She also holds a Masters of Technology and a Bachelor of Engineering degree in Computer Science and Engineering. Her areas of interest are Software Engineering, Requirement Engineering, Software Testing, Software Project Management, Data Structures, Data Mining and Web Applications. She has many publications in international journals and conferences to her credit.

Sangeeta Mittal is Associate Professor in the Department of Computer Science & IT from Jaypee Institute of Information and Technology, Noida, India. She obtained her Doctorate from Jaypee Institute of Information and Technology, Noida. She also holds a Masters of Technology and Bachelor of Computer Science and Engineering. Her areas of interests include Wireless Sensor Networks, Context Aware Systems and Sensor based Smart Environments. She is a member of IEEE, ACM and has many publications in international journals and conferences to her credit.