Social-K: Real-Time K-Anonymity Guarantees for Social Network Applications

Aaron Beach, Mike Gartrell, Richard Han
University of Colorado at Boulder
{aaron.beach, mike.gartrell, richard.han}@colorado.edu

***Abstract*—Traditional approaches to K-anonymity provide privacy guarantees over publicly released data sets with specified quasi-identifiers. However, the most common public releases of personal data are now done through social networks and their APIs, which do not fit the previous research-centric data set release model, nor do they allow for clear assumptions about quasi-identifiers. This paper proposes a new definition of K-anonymity that suggests a practical way in which social networks could provide privacy guarantees to users of their API. To support as wide a range of applications as possible, the proposed privacy guarantee assumes all social-networking data may be a quasi-identifier and does not assume that data may be generalized and still be useful. Using the Facebook social networking API, we implement an application to demonstrate that providing such guarantees in real-time is feasible for real social networking data.**

I. Introduction

Every month at least 250 million Facebook users release their personal social networking data through Facebook's API. The recipients of this data include over 500,000 social networking applications, 80,000 external websites, 65 million mobile devices, and 180 mobile operators [1]. Due to the nature of how this data is used and accessed, the traditional definition of K-anonymity and how it is implemented does not apply to this public release of personal data. This paper suggests an alternative approach to K-anonymity that directly applies to this data and how it is accessed.

This paper introduces a new type of privacy guarantee for online social network (OSN) applications. This privacy guarantee deals with the "re-identification" problem and K-anonymity definition as discussed in [2]. However, this paper suggests a new K-anonymity problem to fit the most common usages of personal social networking data. The traditional approach to K-anonymity is to provide a guarantee over a data set such that quasi-identifiers associated with one identity in the data set are indistinguishable from (copies of) at least $k-1$ other identities in the data set. While interesting as related to data sets for research purposes, the traditional K-anonymity definition is not so useful when applied to the release of social networking data through publicly available social networking APIs for the following reasons:

1) Social networks do not currently anonymize their data sets and have released no plans to do so in the future. Furthermore, the social networks are the only entities capable of such anonymization.
2) It is possible that any or all social network attributes may be used as quasi-identifiers, therefore all social network data (profile data) must be considered as quasi-identifiers.
3) Social network API calls access or refer to only a small subset of the overall data set and usually only refer to a particular subset of attributes related to an individual. Whereas, traditional K-anonymity guarantees require anonymity across the entire data set.

In order for users to maintain anonymity, social networks would have to provide API calls which allow queries to specify individuals with non-unique or anonymous information since knowledge of a users unique identifier is a violation of anonymity. Therefore, this paper argues that social network APIs should either provide a trusted system for generation of anonymous identifiers or support conditional queries which are submitted to the same anonymity guarantees as the personal data released in their responses. Also, this paper assumes that the practicality of social networks changing their existing APIs or providing an API with anonymity guarantees is more reasonable than expecting social networks to anonymize their entire data set.

Therefore, we propose a new but related K-anonymity problem defined as follows:

Given a partial release of data from a personal data set, wherein all data is quasi-identifiable, the released data must map to at least k distinct sets of individuals within the data set.

The primary differences between this definition of K-anonymity and the traditional definition given in [3] are listed below:

All data is considered a "quasi-identifier". This paper takes a "guilty until proven innocent" approach to whether or not data could be used to re-identify an individual. Current social networks such as Facebook have differing and varying access control policies often based on changing social relations or personally defined settings. Therefore, this paper makes the assumption that all social network data may be a quasi-identifier (or cannot be assumed immune to external re-identification attacks) and as such all data are considered quasi-identifiers.

The privacy guarantee describes how a subset of data relates to the entire data set. There is no privacy guarantee on the entire data set and the privacy guarantee only applies to the data specified in a particular release. If

this privacy guarantee were to be extended over multiple releases it would require that all data cumulatively released be evaluated together. When applied to the entire data set the proposed K-anonymity definition is equivalent to the traditional one.

The privacy guarantee refers to a number of sets of individuals rather than a number of individuals. Since an API call may specify a set of users and and not just a single user, we suggest expressing the privacy guarantee in terms of a minimum number of sets and not just individuals. Consider that because conditional queries (e.g.,"users with blue eyes") may apply to sets of one or more individuals, expressing the privacy guarantee in terms of the total number of individuals is somewhat misleading as these sets of users may uniquely map to the released set of data (i.e. cumulatively account for the data uniquely). If there are k individuals in a group that uniquely accounts for a released data set, then every individual in the group is uniquely mapped to the conditionals specified in the query. A simple example of this case is given below.

A. Example

Consider four Facebook users who also have Netflix accounts: John, Bill, Joe, and Karen. John has blue eyes and likes the movies "Spiderman", "X-men", and "Superman". John's friends Bill, Joe, and Karen do not have blue eyes and each likes one of the movies. Bill likes "Spiderman", Joe likes "X-men" and Karen likes "Superman". John has a Facebook application which requests the liked movies of individuals with blue eyes. In this case, the set of data being evaluated for public release is John's liked movies [Spiderman, X-men, Superman]. The released data may partially map to any of the users: John, Bill, Joe, or Karen - however, John alone accounts for all three movies, without which all three other users (Bill, Joe, and Karen) are required to account for the entire set of released data. Therefore, an attacker who knew the movie preferences of each user from a de-anonymized Netflix data set could deduce two distinct possibilities: either John has blue eyes or all three other users have blue eyes (not mutually exclusive). It is in this sense that, while there are four individuals possibly associated with the data, expressing the privacy of the data as K-anonymous with $k = 4$ is misleading. Since the set of all possibilities can be simplified to two distinct possibilities, this paper defines the data release in this example as K-anonymous with $k = 2$ as it is stronger and appears the more meaningful guarantee to the authors.

B. Our Contributions

1) We identify that the traditional definition of K-anonymity is not very useful for social networking applications which are arguably the most common case of public releases of personal data.
2) We give a related but different definition of K-anonymity which is useful for providing privacy guarantees that relate to partial data releases through a "personal" social networking API.
3) We identify the requirements for a personal social networking API, primarily being that the API support queries using non-unique identifiers or anonymous identifiers.
4) We present an algorithm to verify privacy guarantees in real-time using existing logic-minimization techniques.
5) Through an initial feasibility study we show that such an approach to privacy guarantees can work for a reasonable number of users and data sizes on Facebook - however, this evaluation does not provide extensive performance analysis, which is currently under development using more mature logic minimization techniques from the circuit optimization community.

C. Social Networks & Applications

The growth of social networks and the use of public APIs for accessing social network data has created a vast amount of personal information that drives new types of applications. Many of these applications now take into account location and other contextual information to complement the social relations and personal information provided by online social networks. The association of real-time physical information with social network identities highlights just how personal or intimate the data used by social network applications can be, including real-time association of the details of physical contact with one's social network profile [4].

This paper uses a context-aware mobile application for proof-of-concept to emphasize the importance of privacy in emerging mobile social networks and to show that a context-aware mobile social networking application can function without knowing the identities of its users. Types of mobile applications that might function anonymously include: CenceMe, which sends context information to the social network, e.g. the location of the user and contextual cues such as whether the user is talking [5]; Serendipity [6] and WhozThat [7], which both import social context into the local context using mobile devices; and commercial location-aware mobile social networking services such as Brightkite and Loopt.

D. Privacy in Social Networks

Previous work at Duke University [8] has dealt with privacy and anonymity questions as they apply to sharing presence information with other users and matching users with a shared location and time. This work may be used to support anonymous identifiers that would allow anonymous queries for specific users' information. For instance, SmokeScreen [8] presents a protocol by which devices may broadcast identifiers which can be resolved to an identity

through a trusted broker system. However, SmokeScreen does not address the K-anonymity problem that we discuss in this paper.

E. K-anonymity in Social Networks

Prior work on K-anonymity in social networks has largely focused on developing algorithms that anonymize only the social graph of friendships [9], [10], [11], or both friendship and user profile data obtained from social networks [12]. [9], [10], [11] primarily involve perturbation of the social graph structure in a social network. In contrast to this work, Social-K avoids distorting the social graph structure and instead employs a fundamentally different approach that *selectively withholds* user profile data to achieve anonymization such as that discussed in [13]. [12] uses generalization to anonymize user profile data, which consists of replacing the actual value of a data item with a more general value that is considered "faithful" to the original. For example, the value of "80305" for a ZIP code could be generalized to "803**" or "Boulder, CO". Instead of distorting user profile data by generalization, Social-K preserves the accuracy of data by satisfying the K-anonymity guarantee through selective withholding of released data.

II. The Indirect or K-Anonymity Problem

This paper proposes a personal social networking API, that is, an API which provides privacy guarantees for the data that it releases. In particular, it guarantees than an "anonymous" release of personal data cannot be used to distinguish between a minimum number of users, designated by k. The value of k refers specifically to the number of *distinct* and indistinguishable sets of users. This paper chooses to define k in terms of sets of users rather than total number of users due to the latter being implied by the former but not vice versa. An example of this was discussed in section I-A.

This new definition of K-anonymity was motivated by the fact that the existing definition relates to the public release of data sets, usually for research purposes. However, this definition makes assumptions that are not realistic or useful when applied to the how social networks release their personal data. Social networks release their data through public APIs, through which social network applications, external web sites, or mobile applications can integrate social networking data.

The new definition of K-anonymity targets the way in which social networks release their data through APIs. When this problem is stated in set theory it is clear that it is a generalization of the traditional data set release problem assumed by the traditional definition of K-anonymity. A formal definition of this new K-anonymity is given below in terms of sets, however its basic relationship to the traditional definition of K-anonymity is as follows: In the traditional definition of K-anonymity an individual maps to set of quasi-identifiers, and this set of quasi-identifiers is guaranteed to fully map to at least $k-1$ other individuals within the data set. In the new definition of K-anonymity any set of quasi-identifiers which may be released must fully map to at least k distinct sets of users within the data set. Defining the guarantee more generally in terms of all quasi-identifier subsets allows the guarantee to be applied to APIs within which all data is treated as a quasi-identifier and any subset of that data may be queried. Furthermore, data which does not meet the privacy guarantee is withheld. While it may be possible that certain data may be generalized to achieve anonymity and used by some applications, it can not be assumed that this approach applies to all types of applications. Such an approach is considered a separate problem beyond the scope of this paper.

Definition 1. K-Anonymity
Given two sets U and T. U is a set of all individuals and T is a set of all quasi-identifiers. There is a many-to-many mapping between T and U. t is any subset of T and u is a subset of U. T_k is the superset of all sets t_x which may be released under a K-anonymity guarantee, then
$\forall t_x \in T_k, \exists\{u_1, \ldots, u_k\} \in U_k$ where: $\forall u_x \in U_k, u_x \mapsto t_x$ and $|U_k| = k$.

A. Note on the relation to Logic Simplification

The distinct sets of individuals which map to a set of quasi-identifiers can be expressed as distinct (or disjunct) possibilities in a Boolean algebra equation. The equation would take a sum-of-products form in which each set of individual(s) is expressed as a disjunct clause of literals, where each individual is its own literal and each clause is a set of individuals which fully maps to the set of quasi-identifiers. Expressing the sets of individuals this way allows us to find the "minimal" number of sets which could possibly account for the set of quasi-identifiers.

Refer back to the story at the end of the introduction in which an attacker was able to use known movie preferences to deduce a minimum of two possibilities regarding the unknown eye color of the individuals. The attacker could in fact have considered many more redundant possibilities, such as the set of all individuals or any set that included the one individual which fully accounted for the entire set of quasi-identifiers. As such, it is necessary to find the "minimal" set of possibilities to provide a true K-anonymity guarantee.

To find this minimal set of possibilities we express the possible sets of individuals as a Boolean algebra equation and find the minimal number of prime implicants of the expression. This can be done by finding the minimal disjunctive normal form (DNF) expression. This is the problem solved by the classic Quine-McCluskey algorithm[14] and many other logic minimization techniques. This equivalence allows us to pose the problem of guaranteeing K-anonymity as a Boolean algebra or logic minimization problem and in

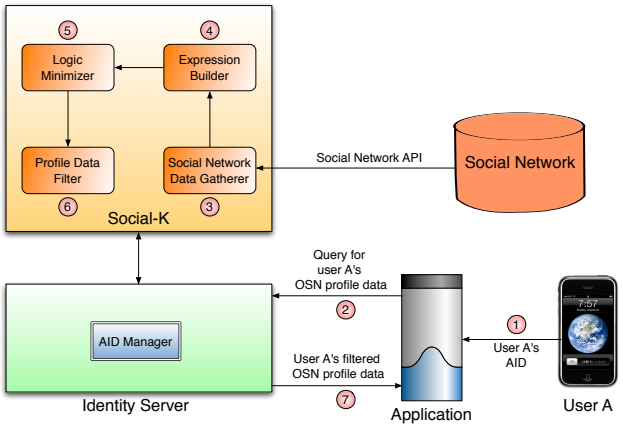

Figure 1. Interaction between Social-K and a social network application

doing so, allows us to draw upon the wealth of research into practical, fast, and efficient solutions to this problem, such as ESPRESSO [15].

III. Social-K Architecture

This section will outline the architecture of an example personal social network API, Social-K. Social-K was implemented on top of the Facebook API to test feasibility of such a solution. The system is made up of five major components: Identity Server, Social Network Data Gatherer, Expression Builder, Logic Minimizer, and Profile Data Filter, as shown in Figure 1. All of these components are centralized and trusted. The Identity Server manages mappings between unique identifiers and anonymous sessions or nonce values which are generated by the Identity Server and used by applications to make anonymous API calls to the system. The Social Network Data Gatherer processes Social-K API calls by retrieving, filtering, and appropriately storing the necessary social network data for the other components. The Expression Builder takes the non-anonymized data from the social network and converts it into a Boolean expression as discussed in section II-A. The Logic Minimizer simplifies the expression from the Expression Builder, producing a simplified DNF expression that can be evaluated under the K-anonymity guarantee. The Profile Data Filter returns the results to a query if the Logic Minimizer produces an expression with at least K disjunctive clauses guaranteeing the K-anonymity of the response, otherwise it removes/selectively withholds values as necessary to meet the K-anonymity guarantee.

The Facebook API is not only used within the context of browsing Facebook.com, but is also accessed from over 80,000 external web sites and 65 million mobile devices. Our main interest in the Facebook API has been related to mobile applications and as such, the example application was chosen to be a context-aware mobile application which uses the Facebook API to request movie preferences of individuals.

Social-K was implemented on top of Facebook which means that our service has only partial access to the entire social network. However, the system is still capable of analyzing parts of the network which are sufficient to provide network-wide anonymity guarantees as long as at least k distinct sets of users are found. If such a system were implemented directly by a social network, such as Facebook, it would not require a data gathering layer as it could access the database directly and could perform more efficient queries against the entire social network.

Figure 1 depicts Social-K processing a query for a user's Facebook profile data. This figure also shows how each of the Social-K components interact in fulfilling the query.

A. Identity Server

The Identity Server (IS) provides identity management services for social network application users. The IS generates and distributes anonymous identifiers (AIDs) to users through requests. An AID is a nonce which may be included with a query to anonymously identify the user (or users) associated with the query. A user's AID is mapped to the user's social network ID by the database on the IS.

Whenever a user seeks to advertise his or her AID to an application, he/she first requests an AID from the IS. The IS generates a new AID using a cryptographic hash function such as SHA-1, with a random salt value. The IS associates the newly generated AID with the user and returns the new AID. AIDs may then be shared safely with applications, which may be social network applications running in a Web browser, on a phone, or even integrated into a context-aware application as described in section IV.

B. Social Network Data Gatherer

After the IS receives a request for a user's social network profile data (we will call this user "user A") Social-K begins gathering pertinent user profile data for a large set of users. It was convenient to start with the user's friends on the social network since they were easiest to access through the Facebook API.

C. Expression Builder

Upon retrieving the relevant user profile data for a set of users, Social-K invokes the Expression Builder component to construct a Boolean expression representing the relationship between all of the users and their profile information. The following describes an example of how individuals and profile data being considered for release can be combined into a Boolean expression:

Consider the example shown in Figure 2. If the data set *(Chemistry class, Anne, 1)* is released, it could be mapped back to the distinct sets $(Bill)$ and $(Fred, Joe)$ implying that at least, Bill **OR** (Fred **AND** Joe) could have generated

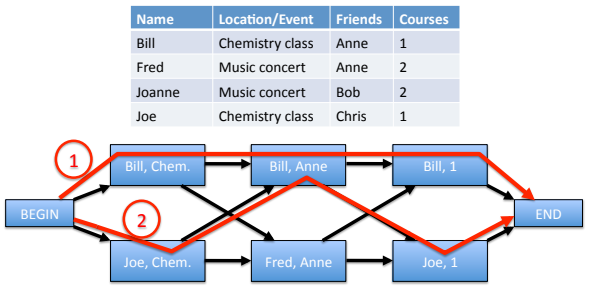

Figure 2. Example of how released data set {Chemistry Class, Anne, 1} and its associated individuals can be expressed as a directed graph in which all paths are sets of individuals which could possibly account for the released data set

the data. This would be an example of K-anonymity where $K \leq 2$.

For more complex data sets, the Boolean expression linking the set of users to the set of data to be released needs to be systematically derived, which can be achieved using a directed graph that models the relationship between the data to be released and the individuals who could be linked to that data. For any data set that we wish to release (d_1, d_2, ..., d_n), we construct a column of nodes for each d_i, where each node in column i consists of the pair (username, d_i). This identifies all possible users who could be associated with the release of data item d_i. Next, we interconnect the nodes in a column with the nodes in the next column. This creates the directed graph. The set of all truth cases would be the superset of all paths across the graph. Each path would map to a conjunctive clause of literals (one literal per node) in the final disjunctive normal form (DNF) Boolean expression.

For example, before releasing *(Chemistry class, Anne, 1)*, we generate the directed graph shown in figure 2 consisting of three columns, one each for all users associated with the Chemistry class, Anne, and one course. A single path #1 through the graph corresponds to the Boolean expression (Bill AND Bill AND Bill), or just (Bill). A second zigzag path #2 corresponds to the Boolean expression (Joe AND Bill AND Joe), or just (Joe, Bill). The union of all possible paths through the graph gives us all possible combinations of users that could be associated with the release of the data *(Chemistry class, Anne, 1)*.

D. Logic Minimizer

Next, we apply logic minimization algorithms to simplify the Boolean expression provided by the Expression Builder. Several well known logic minimization algorithms exist, including ESPRESSO [15] and Quine-McCluskey [14]. When applied to our example graph in figure 2, the simplified Boolean expression reduces to (Bill) OR (Joe, Fred). If the number of disjunct clauses is greater than or equal to k then the data is admissible under a K-anonymity guarantee, if not the data must be filtered by the Profile Data Filter.

E. Profile Data Filter

If a set of data is found to not meet the required K-anonymity guarantee then the data is reduced and tested again. This project has not yet designed any advanced technique beyond random withholding and retest, however any mature solution to this problem should consider this step seriously as it involves a trade-off between response time, computational resources, and information loss. Based on the data in figure 2 consider the simple example below of how withholding a piece of data may increase the value of k.

If we wish to release *(Chemistry class, Chris, 1)*, then a single possibility may be deduced - that Joe is related to the data since he is the only friend of Chris. However, if we withhold the data item *(Chris)* and only release *(Chemistry class, 1)*, then the possible identities are *(Joe)* OR *(Bill)*. Thus, we've increased the anonymity to $K \leq 2$.

IV. Feasibility Study

We have implemented a prototype of Social-K and performed an initial evaluation of its behavior and feasibility. This section describes our implementation and some initial results from a feasibility study.

A. Implementation

The IS was implemented using the Java Standard Edition (SE) 5.0 platform. All IS services accessed by mobile and/or stationary devices are exposed as web services conforming to the REST architecture [16]. We expose each resource on the IS, including a user's AID and the Facebook profile information for a user, as separate URL-accessible resources supporting the HTTP GET method. The body of each HTTP request is encoded using JSON (RFC 4627). All web service network traffic between the IS and other mobile/stationary devices is encrypted using HTTPS, and access to all resources is authenticated using HTTP basic access authentication (RFC 2617). [17] provides more information about the implementation of the IS.

We use an open source Quine-McCluskey implementation [18] to perform logic minimization in Social-K. As our evaluation results will show, this component performs reasonably well as the number of variables and terms in the Boolean expression to be minimized increases.

We use the SocialAwareFlicks application described in [19] as an example of a context-aware social networking application that queries the IS for user profile data. SocialAwareFlicks displays movie trailers that match the movie preferences of one or more users jointly watching a common large-screen display.

B. Feasibility Study Results

We have gathered some initial performance metrics to demonstrate the feasibility of Social-K. All performance testing was performed using a Macbook notebook running Mac OS X 10.5, with a 2.0 GHz Core2Duo processor, 2 GB of RAM, and a university-provided high-speed Internet connection.

Our metrics were gathered using the Facebook account of a volunteer, here called "user A", who has 222 Facebook friends and seven favorite movies listed on his Facebook profile. In a study conducted by Ellison et al. [20], the mean number of Facebook friends reported by the study participants was between 150 and 200. Therefore, we suppose that user A is a reasonable representation of a typical or average Facebook user.

We conducted our evaluation of Social-K performance by submitting a query to the IS requesting the list of favorite movies in the Facebook profile for user A. Social-K begins processing this query by first gathering the favorite movies lists for each of user A's Facebook friends. Social-K then proceeds to use the Expression Builder component to construct a Boolean expression representing the relationship between user A's friends and their favorite movies.

Figure 3 shows how the time required to minimize the unsimplified Boolean expression in the Social-K Logic Minimizer component varies with the number of terms in the unsimplified Boolean expression. We see from this plot that the Logic Minimizer component scales reasonably well for unsimplified Boolean expressions containing up to about 450 terms. We expect that 450-term Boolean expressions will account for many typical usage scenarios, although this will vary based on the number of the user's favorite movies that match with his friends' favorite movies. There is a nonlinear relationship between the number of terms in the unsimplified Boolean expression and the number of terms in the simplified expression. Based on the results of our tests, we have found that unsimplified Boolean expressions containing around 450 terms have up to about 20 terms when simplified by the Social-K logic minimizer. 20 terms in the simplified Boolean expression provides K-anonymity guarantees for $k = 20$. Thus, we have shown that Social-K is feasible for K-anonymity guarantees up to $k = 20$, which includes user groups as large as most social network friend lists (consisting of 200–300 friends).

Figure 4 shows how the time required to minimize the unsimplified Boolean expression in the Social-K Logic Minimizer component varies with the number of terms in the simplified Boolean expression (k). We see from this plot that there is minimal correlation between the value of k and the time required to minimize the unsimplified expression. We can conclude from figures 3 and 4 that the time to minimize the unsimplified Boolean expression is correlated with the size of the input to the Social-K Logic Minimizer component

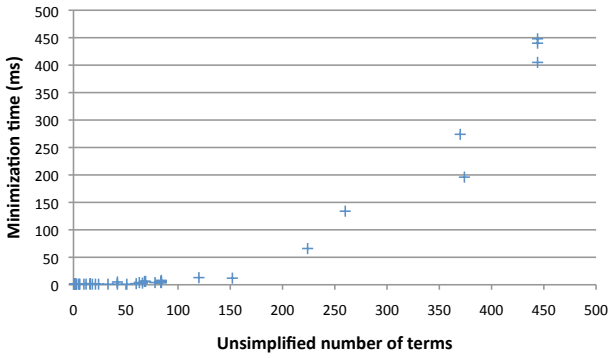

Figure 3. Time to minimize the unsimplified Boolean expression vs. number of terms in the unsimplified Boolean expression

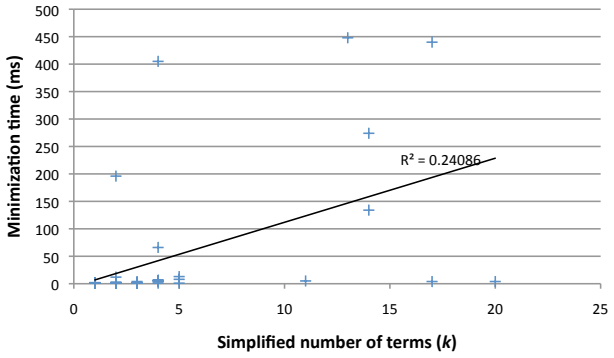

Figure 4. Time to minimize the unsimplified Boolean expression vs. number of terms in the simplified Boolean expression (k)

(the number of terms in the unsimplified expression), and is not correlated with the size of the output (the number of terms in the simplified expression). This is the expected behavior for a logic minimizer.

Table I shows the run times of each component of Social-K for the following conditions for user A:

- Number of friends: 222
- Number of movie matches: 13
- Number of friend matches: 7
- Number of terms in unsimplified expression: 339
- Number of terms in simplified expression: 7

The Social-K total run time in table I is the time for our system to return K-anonymous favorite movie preferences for a user. In our tests, the run time of the Social Network Data Gatherer component dominates the run time of Social-K, since this component spends most of its time downloading data from Facebook. We expect that running Social-K with local access to Facebook's user database would significantly reduce the run time of this component. However, the current average total Social-K run time of 1377 ms for our tests provides acceptable performance for applications such as SocialAwareFlicks.

Component	Mean run time (ms)	Run time standard error (ms)
Social network data gatherer	852	145
Expression builder	11	1.7
Logic minimizer	297	8.29
Social-K total	1377	134.7

Table I
RUN TIMES FOR EACH SOCIAL-K COMPONENT

V. CONCLUSION

This paper presents Social-K, a new approach to K-anonymizing social network data whereby data is released without modification as long as K-anonymity constraints are met, and is otherwise selectively withheld. This contrasts to existing approaches that release modified data, either distorted or generalized, to maintain K-anonymity. Social-K further offers a new and useful definition of K-anonymity in relation to social network queries. This privacy guarantee is defined in terms of set theory, which relates sets of users to sets of data, thereby allowing the problem to be solved using equivalent logic minimization algorithms, for which many efficient solutions exist. This Social-K solution is then tested with a proof-of-concept implementation which demonstrates that it is practical to employ our logic minimization approach to K-anonymize social networking data profiles.

REFERENCES

[1] "Facebook statistics," http://www.facebook.com/press/info.php?statistics.

[2] L. Sweeney, "Uniqueness of simple demographics in the U.S. population," in *LIDAPWP4*, 2000.

[3] ——, "k-anonymity: a model for protecting privacy," *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, vol. 10, no. 5, pp. 557–570, 2002.

[4] A. Beach, B. Ray, and L. Buechley, "Touch me wear: Getting physical with social networks," *Workshop on Social Computing with Mobile Phones & Sensors: Modeling, Sensing and Sharing (SCMPS09) at SocialCom09 in IEEE International Conference on Computational Science and Engineering*, vol. 4, pp. 960–965, 2009.

[5] E. Miluzzo, N. D. Lane, S. B. Eisenman, and A. T. Campbell, "Cenceme - injecting sensing presence into social networking applications," in *Proceedings of the 2nd European Conference on Smart Sensing and Context (EuroSSC 2007)*, October 2007, pp. 1–28.

[6] N. Eagle and A. Pentland, "Social serendipity: Mobilizing social software," *IEEE Pervasive Computing*, vol. 4, no. 2, pp. 28–34, 2005.

[7] A. Beach, M. Gartrell, S. Akkala, J. Elston, J. Kelley, K. Nishimoto, B. Ray, S. Razgulin, K. Sundaresan, B. Surendar, M. Terada, and R. Han, "Whozthat? evolving an ecosystem for context-aware mobile social networks," *IEEE Network*, vol. 22, no. 4, pp. 50–55, July-August 2008.

[8] L. P. Cox, A. Dalton, and V. Marupadi, "Smokescreen: flexible privacy controls for presence-sharing," in *MobiSys '07: Proceedings of the 5th international conference on Mobile systems, applications and services*. New York, NY, USA: ACM, 2007, pp. 233–245.

[9] G. Miklau, D. Jensen, P. Weis, and S. Srivastava, "Anonymizing social networks," Computer Science Department, University of Massachusetts Amherst, Tech. Rep. Technical Report 07-19, March 2007.

[10] Q. Wei and Y. Lu, "Preservation of privacy in publishing social network data," in *Proceedings of the 2008 International Symposium on Electronic Commerce and Security (ISECS 2008)*. IEEE Computer Society, March 2008, pp. 421–425.

[11] B. Thompson and D. Yao, "The union-split algorithm and cluster-based anonymization of social networks," in *Proceedings of the 4th International Symposium on Information, Computer, and Communications Security (ASIACCS 2009)*. ACM, March 2009, pp. 218–227.

[12] A. Campan and T. Truta, "A clustering approach for data and structural anonymity in social networks," in *PinKDD 2008*. ACM, August 2008.

[13] C. D. Capitani, V. Ciriani, S. De, C. Vimercati, S. Foresti, and P. Samarati, "k-anonymity," *Secure Data Management in Decentralized System*, 2007.

[14] S. Muroga, *Logic Design and Switching Theory*. New York: Wiley, 1979.

[15] R. L. Rudell, "Multiple-valued logic minimization for pla synthesis," EECS Department, University of California, Berkeley, Tech. Rep. UCB/ERL M86/65, 1986. [Online]. Available: http://www.eecs.berkeley.edu/Pubs/TechRpts/1986/734.html

[16] R. Fielding, "Representational state transfer (rest)," http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm.

[17] A. Beach, M. Gartrell, and R. Han, "Solutions to security and privacy issues in mobile social networking," in *SMW09: Workshop on Social Mobile Web at SocialCom 2009 in CSE '09: Proceedings of the 2009 International Conference on Computational Science and Engineering*. Washington, DC, USA: IEEE Computer Society, 2009, pp. 1036–1042.

[18] "Quine-mccluskey algorithm (java)," http://en.literateprograms.org/Quine-McCluskey_algorithm_(Java).

[19] C. M. Gartrell, "Socialaware: Context-aware multimedia presentation via mobile social networks," Master's thesis, University of Colorado at Boulder, December 2008, http://www.cs.colorado.edu/~rhan/Papers/Mike_Gartrell_CU_MS_thesis-final.pdf.

[20] N. Ellison, C. Steinfield, and C. Lampe, "The benefits of facebook "friends:" social capital and college students' use of online social network sites," *Journal of Computer-Mediated Communication*, vol. 12, no. 4, 2007, http://jcmc.indiana.edu/vol12/issue4/ellison.html.