On query self-submission in peer-to-peer user-private information retrieval

Klara Stokes
Universitat Rovira i Virgili
Dept. of Computer Engineering and Maths
UNESCO Chair in Data Privacy
Av. Països Catalans 26
43007 Tarragona, Catalonia, Spain
klara.stokes@urv.cat

Maria Bras-Amorós
Universitat Rovira i Virgili
Dept. of Computer Engineering and Maths
UNESCO Chair in Data Privacy
Av. Països Catalans 26
43007 Tarragona, Catalonia, Spain
maria.bras@urv.cat

ABSTRACT

User-private information retrieval (UPIR) is the art of retrieving information without telling the information holder who you are. UPIR is sometimes called anonymous keyword search. This article discusses a UPIR protocol in which the users form a peer-to-peer network over which they collaborate in protecting the privacy of each other. The protocol is known as P2P UPIR. It will be explained why the P2P UPIR protocol may have a flaw in the protection of the privacy of the client in front of the server. Two alternative variations of the protocols are discussed. One of these will prove to resolve the privacy flaw discovered in the original protocol. Hence the aim of this article is to propose a modification of the P2P UPIR protocol. It is justified why the projective planes are still the optimal configurations for P2P UPIR for the modified protocol.

Categories and Subject Descriptors

H.2.7 [**Database management**]: Database administration—*Security, integrity and protection*

General Terms

Security

Keywords

User-private information retrieval, anonymous keyword search, cryptographic key distribution, block designs, combinatorial configurations

1. INTRODUCTION

Privacy is the ability to choose to reveal to others only what you want to reveal. Private information retrieval (PIR) is the art of retrieving information without telling the information holder *what* information is retrieved [3]. By protecting the identity of the information retriever instead of the identity of the retrieved data, we get what is called user-private information retrieval (UPIR) [6]. UPIR is the art of retrieving information without telling the information holder *who* you are. UPIR is sometimes called *anonymous keyword search*. The information holder is typically a server.

Advantages of the UPIR protocol over existing PIR protocols are for example that UPIR does not need cooperation from the server, as some PIR protocol do. that UPIR can obtain sublinear complexity, compared to the linear complexity of the best PIR protocols and that existing PIR protocols usually model the database as a vector, but for UPIR this is not needed. In [5], a UPIR protocol was presented which was based on a peer-to-peer network, P2P UPIR. The idea behind the P2P UPIR protocol is that the clients who want to retrieve information collaborate in posting each others queries. The clients use a P2P network to interchange the queries and the answers to the queries. In addition to preserving the privacy of a user's query profile in front of the database and external intruders, P2P UPIR offers privacy versus peer users. Other users see only a small part of the other user's queries. Peers can be made anonymous to each other also on the network layer by using mixers.

The communication over the P2P network should be encrypted. We assume that the encryption is done using a symmetric encryption scheme. If the encryption is made with the same key over the entire network, then there is a high risk that the key is compromised. On the other hand, if the encryption uses different keys for every pair of clients, then this risk is low. But if the protocol prescribes one key for every pair of clients and the number of clients is large then the number of needed keys is very large, which is a problem. There are however more sophisticated ways to distribute cryptographic keys than the two trivial examples just described. The articles [5, 6] treat a version of the P2P UPIR protocol which uses combinatorial configurations (defined below) to manage the keys. The idea to use combinatorial configurations for key distributions can also be found in [9]. Previous articles have treated the existence and the constructions of configurations for P2P UPIR [2, 11]. Some questions regarding optimality in privacy and efficiency for P2P UPIR have been treated in [10] (see below). Versions of P2P UPIR which do not use combinatorial configurations can be found in [12, 4].

A combinatorial (v, b, r, k)-configuration is a nonempty incidence structure, that is, a triple $\mathcal{S} = (\mathcal{P}, \mathcal{L}, \mathcal{I})$, where $\mathcal{P}$ is a nonempty set of "points", $\mathcal{L}$ is a nonempty set of "lines", and $\mathcal{I} \subset (\mathcal{P} \times \mathcal{L}) \cup (\mathcal{L} \times \mathcal{P})$ is a symmetric incidence relation, such that each point is on $r \geq 2$ lines, each line has $k \geq 2$ points and any two different points are incident with at most one line, or equivalently, any two different lines are incident with at most one point. The meaning of the first two parameters is $v := |\mathcal{P}|$ and

PAIS 11 March 25, 2011, Uppsala, Sweden

$b := |\mathcal{L}|$. No geometric meaning is attached to the terms point and line. A line is simply a subset of cardinality k of the set of points. We say that a point p is incident with a line l if $(p, l) \in I$, that is, if p is a point on l. A general reference for combinatorial configurations is [7] and the recently published [8] collects many results on combinatorial configurations, although it focuses on geometrically realizable configurations.

The idea behind the key distribution used in [5, 6] is to represent the collaborating clients by the points of a combinatorial configuration and to use the lines to represent "communication spaces", that is, a memory sector together with a belonging cryptographic key. A client represented by the point p has access to the communication spaces which are represented by the lines through p and he stores the keys corresponding to these communication spaces. When the client wants to submit a query to the server, then he uploads the query to one of the communication spaces to which he has access, after encrypting it with the corresponding cryptographic key. Another client represented by the point q can read p's query on the communication space iff he has access to the corresponding cryptographic key. In other words, the client q can read p's query iff the communication space is represented by a line passing through both p and q.

Next, the client q posts the query to the server. When q receives the answer to the query he uploads it to the same communication space from where he previously read the query, after encrypting it with the corresponding cryptographic key. Subsequently p can read the answer to his query from the communication space, after decrypting it.

Below we present the configuration based P2P UPIR protocol as described in [5, 6]. The precondition of the protocol is that the client or user wants to post a query to the server. The postcondition of the protocol is that the client or user obtains the answer to his query. We will abuse notation and not distinguish the points and the lines of the configuration from the clients and the communication spaces which they represent.

PROTOCOL 1 (P2P UPIR (I)). *1. A client or user represented by the point u selects randomly a communication space represented by a line c passing through u.*

2. u decrypts the content on the memory sector of c using the corresponding cryptographic key of a symmetric cipher. Now the protocol ramifies into five cases depending on the outcome of the decryption.

*(a) The outcome is **garbage**. Then u encrypts his query and records it in c;*

*(b) The outcome is **a query posted by another user**. Then u forwards the query to the server and awaits the answer. When u receives the answer, he encrypts it and records it in c. He then restarts the protocol with the intention to post his query;*

*(c) The outcome is **a query posted by the user himself**. Then u does not forward the query to the server. Instead u restarts the protocol with the intention to post his query;*

*(d) The outcome is **an answer to a query posted by another user**. Then u restarts the protocol with the intention to post his query;*

*(e) The outcome is **an answer to a query posted by the user himself**. Then u reads the query and erases it from the communication space. Subsequently u encrypts his new query and records it in c.*

In [10] it was proved that the finite projective planes are optimal configurations for P2P UPIR (I), after taking into account the privacy of the user in front of the server and in front of other users and also the efficiency. A finite projective plane of order $r - 1$ is a combinatorial $(r^2 - r + 1, r^2 - r + 1, r, r)$-configuration. One can also define a finite projective plane as an incidence structure $\mathcal{S} = (\mathcal{P}, \mathcal{L}, \mathcal{I})$, such that each pair of lines meet in exactly one point, each pair of points span exactly one line and there are four points such that no line is incident with more than two of them. In a finite projective plane any two different points are incident with *exactly* one line, or equivalently, any two different lines are incident with *exactly* one point.

In a finite projective plane every two points are on a triangle. A triangle is a collection of three lines intersecting pairwisely in three distinct points. It was observed in [11] that the existence of triangles in the configuration used for P2P UPIR (I) makes collusions possible where pairs of users with a common neighbor can spy on this neighbor. The collusion consists in that the two users interchange the information they have on the third user's profile over the channel provided by the third edge in the triangle. Therefore, the use of triangle-free combinatorial configurations was proposed for P2P UPIR in [11].

This article mainly focuses on steps 2.(b) and 2.(c) of the protocol above. It will be explained why the P2P UPIR (I) protocol may have a flaw in the protection of the privacy of the client in front of the server in the case that a user submits many repetitions of a rare query and none of those repeated queries takes too long to be processed by another user. Two alternative protocols will be discussed. The first, P2P UPIR (II), is in some sense the opposite of P2P UPIR (I) and it will be proved that also this version of the protocol has a serious flaw in the protection of the privacy of the client in front of the server. The second new protocol, P2P UPIR (III), which is a compromise between the other two protocols, will be proved to resolve the privacy flaw discovered in P2P UPIR (I).

The rest of this article is organized as follows. Section 2 contains a description of an attack on the P2P UPIR (I) protocol. It also contains some real examples from the AOL query logs in order to provide some real illustration of the problem. Section 3 contains the definition and analysis of the two alternative protocols. In Section 4 conclusions are drawn.

2. AN ANALYSIS OF THE P2P UPIR (I) PROTOCOL

The purpose with the P2P UPIR protocol is to protect the privacy of the user when retrieving information from a database from a server. Therefore the natural starting point for the analysis is the privacy of the user in front of the server. We start this section defining some concepts which will be used in the analysis.

DEFINITION 1. *We call the collection of queries which the user u posts to the communication spaces which are incident with him* the real profile of u.

DEFINITION 2. *We call the collection of queries which the user u posts to the server* the apparent profile of u.

DEFINITION 3. *We call the collection of users which are collinear with a user u but different from u* the neighbors of u *and denote these by $N(u)$.*

In the P2P UPIR (I) protocol the user forwards to the server only queries from collinear users different from himself. This strategy is controlled by steps 2.(b) and 2.(c) in the protocol. We will now

see that it is not the perfect strategy to follow. Rather it causes the user to put his privacy at risk.

In any combinatorial configuration, given a point u the number of lines through u is r and the number of points on any of these lines is k (counting u). Therefore the number of neighbors of u is always $|N(u)| = r(k-1)$. Also, in a combinatorial configuration we have the following well-known bound on the number of points v, easily proved by fixing a point and counting the number of points which are collinear with that point.

PROPOSITION 4. *In a combinatorial (v, b, r, k)-configuration we always have*

$$v \geq r(k-1) + 1.$$

A combinatorial (v, b, r, k)-configuration satisfying the inequality in Proposition 4 with equality is a finite projective plane of order $r - 1 = k - 1$. In other words, for a combinatorial configuration which is a finite projective plane of order $r - 1 = k - 1$ we have $r(k-1) + 1 = v$, where v is the total number of points. Consequently, for any point u in the finite projective plane, since the number of neighbors is $|N(u)| = r(k-1)$, the neighbors of u must be all points except u. In particular, in this case, given u, $N(u)$ is always trivially known. Therefore, if u posts repeatedly a unique query, then the server can deduce that u is posting the query since he is the only user not posting the query. From this perspective, a finite projective plane therefore seems to be a bad choice of configuration for the P2P UPIR (I). Using a finite projective plane implies that any repeated query odd enough to identify u can be traced back to him. The reason for this is that *u is the only user not posting his queries to the server*.

The P2P UPIR (I) protocol is designed to protect, for example, the privacy of users of web-based search engines. We say that a profile is rare if it contains many unique queries or unique combinations of queries and we say that it has repetition if it contains many repeated queries or repeated variations of queries. The scope of this article is to investigate if reidentification is possible considering the worst case scenario, that is, when the profile of a user is rare and has repetition. We have seen above that a user with a worst case scenario profile is vulnerable for reidentification attacks when the configuration used is a finite projective plane. This contradicts the recommendation from [10] to use finite projective planes for P2P UPIR (I). However, one should notice that in this analysis we assume that the configuration which is used in the protocol is secret. This can of course not be assumed. Indeed one should always assume that everything in the protocol except for the cryptographic keys is public knowledge. Also, it would surely be very inefficient to construct a new configuration (the topology of the P2P network) for every collection of users that wants to implement the protocol. Finally, if it is decided that the configurations to use should be finite projective planes, then it must be taken into account that there are very few such planes for a given number of users, so in this case there is no secret at all or hardly any secret at all.

We assume the Kerckhoffs' principle and so we assume that the topology of the configuration is public. Then $N(u)$ is known for all u. Therefore, given a collection of users X with an apparent profile containing some rare repeated query, it should be possible to identify a (short) list of users u_i such that $N(u_i)$ is approximately X. In this way it should be possible to reidentify the owner of the real profile behind the queries in the collection of apparent profiles, at least partially. From this perspective the finite projective plane are indeed the optimal configurations for P2P UPIR (I). The reason for this is that in a finite projective plane the real profiles of the users are maximally dispersed by P2P UPIR (I), that is, distributed into a maximal number of apparent profiles. What we saw in the previous analysis was however that the diffusion performed by the P2P UPIR (I) can never be complete. Concluding, a worst case scenario real profile can be mapped to the user behind this profile, also when the configuration which is used is optimal!

One can argue that in the description of the P2P UPIR (I) in [5, 6] the protocol lets the user post his own queries if the waiting time for another user to post it is too long. Therefore it is of course possible that the user by accident is lucky enough to posts the same proportion of his queries as do his neighbors, meaning that in this case the attack described above would not work. One can however not rely on such arbitrary circumstances for the protection of the privacy of the user.

One can ask if it is a common behavior of real users to repeatedly post a query. An interesting question is also how a typical real profile of a user look like. In 2006 AOL released search logs containing 20 million web queries from 658,000 AOL users posted in a period of 3 months. The released data was anonymized by replacing the identity of the users by a random index, but this quickly showed insufficient as several sequences of queries were mapped to real persons. AOL withdrew the query logs from internet, but the files were of course already downloaded by many people. The AOL search data release caused a privacy scandal which is the reason why the query logs published by AOL are practically the only material available for non-corporative research on the subject.

A quick look at the AOL query logs [1] makes it reasonable to assume that posting the same query (or a variation of a query) several times is a common behavior of users of web-based search engines. There seems to be at least three scenarios which can result in this behavior.

The first scenario is due to the way people normally use their browser. In the common internet browsers, when the user queries a search engine the result will be presented to him as a list of links in the browser window. The user's next step is to choose a promising link from the list and to follow it. Later he may want to return to the list of search results. Although he may still have the page with the list of search results open in the browser, this page will not be on top of the windows the user has opened. The user, being lazy, does not change to the previous window with the search results, nor does he press the return button of the browser in order to return to the previous window, but simply posts the query again. This behavior leads to many repeated queries without much or any variation.

In the released AOL query log files there are many query sequences with repeated queries which can be explained by this scenario. For example, user 1783081 has one query for 'digital camoflasges' at 2006-03-15 12:49:29 and then 9 identical queries for 'digital camouflages', the last one posted at 2006-03-15 13:00:09. AOL registered 7 different clicked url as a result from this sequence of queries, giving an example of a user which probably has followed a behavior similar to the one just described. Between 2006-04-18 15:14:03 and 2006-04-18 15:14:03 user 672368 posts 7 queries on 'abortion clinic charlotte' and later between 2006-04-18 21:45:39 and 2006-04-18 21:45:49 5 queries on 'abortion clinic charlotte nc' We observe that some users have sequences with up to 25 equal queries in very short time.

The second scenario is when the user is repeating very similar queries in order to adjust and limit the search result so that it resembles more what the user aimed at. Misspellings is a similar scenario, but misspellings do not tend to result in multiple repetitions of a query. For example, between 2006-03-19 19:24:09 and 2006-03-19 19:30:02 the user 1783081 from the previous examples posted 3 queries on 'the long ranger', 1 query on 'the legend of the long ranger', 5 queries on 'the legend of the lone ranger', 6 queries on 'the lone ranger theme song' 3 queries on 'lone ranger

theme'. User 1783081 generally shows a general interest for fantasy, movies and as more particular interests figures lolita porn, occult rituals, incest and young teen girls. User 672368 has a sequence of queries starting at 2006-04-18 06:50:07 with a query 'effects oon on fibriods', then three queries on 'effects of abortion on fibroids' and four queries on 'abortion fibroids', with the last query at 2006-04-18 06:59:32. After this the user continues to post queries for example on the subject 'abortion'. At 2006-04-20 17:55:18 the user continues posting 11 queries on 'abortion fibroids'. Totally on this subject the user posts 19 queries on the subject 'abortion fibroids'. The user started the query sequence with 'curb morning sickness', 'get fit while pregnant', continued with 'you're pregnant he doesn't want the baby' and many queries on abortion, abortion clinics and later misscarriage. It seems likely that this user would have preferred a better privacy than AOL could offer.

The third scenario is when the user posts queries on something which appears in his daily life. For example, it seems to be rather common that users post a query to a search engine in order to search for the webpage of the school of their kids, or their own workplace, instead of browsing to the webpage directly. The user's workplace and the school of his kids are highly interesting information for reidentification. Considering that this kind of queries can be repeated several times a month, the risk of reidentification can not be neglected.

The AOL query logs are not well suited for finding repeated examples of the third scenario, since they only cover a time period of 3 months. However, AOL themselves and other search engine providers have of course access to query logs from much longer time periods.

It should be observed that although the P2P UPIR (I) protocol fails to provide complete protection of the privacy of the user in front of the server in the case of many repeated queries, single queries can still not be traced back to the emitter.

The level of privacy provided by the P2P UPIR (I) protocol can be specified more exactly. The user diffuses his real profile into the apparent profiles of his $r(k-1)$ neighbors. However since he chooses communication space randomly and has no control over who will forward the query of the other $k-1$ users sharing the same communication space, it is not possible to give more than a statistical hint on how many of the same query a user must post until his privacy is broken. Also, in general it is possible that a user may be the unique common neighbor also to sets of users of cardinality smaller than $r(k-1)$. This also affects the efficiency of the attack.

Finally it should be noticed that this article indeed provides a fix of the problem encountered in the P2P UPIR (I) protocol, as will be seen in the following Section 3.

3. VARIATIONS OF THE PROTOCOL

The previous section was dedicated to a privacy analysis of the P2P UPIR (I) protocol, which is the version of the P2P UPIR protocol appearing in [5, 6]. In the P2P UPIR (I) protocol the user forwards to the server only queries from collinear users different from himself.

In this section we will discuss two variations of the P2P UPIR (I) protocol. The discussion will provide a modification of the protocol which solves the privacy flaw discussed in the previous section.

We define the P2P UPIR (II) protocol as obtained from the P2P UPIR (I) protocol by replacing step 2.(b) and 2.(c) by the single step:

2.(b) The outcome is ***a query posted either by the user himself or by another user****. Then u forwards the query to the server and awaits the answer. When u receives the answer, he encrypts it and records it in c. He then restarts the protocol with the intention to post his query;*

Hence, the only difference from the P2P UPIR (I) protocol is that in the latter the user does not forward his own queries to the server, but in the P2P UPIR (II) he does.

The following proposition shows that the users in a community following the P2P UPIR (II) protocol will forward more of their own queries to the server than of the queries of the other users. As a consequence of this, the users' real profiles can be inferred from the apparent profiles of the users.

PROPOSITION 5. *Consider a community of users $\{u_i\}$ implementing the P2P UPIR (II) protocol. Suppose that in a fixed time interval t a user u_i posts q_i queries. Denote by p_{ij} the proportion of queries from the real profile of u_i on the communication space c_j. Let $\{u_{ij_n}\}$ be the set of communication spaces incident with u_i, indexed by $n \in [1, \ldots, r]$. Then the proportion of queries from the real profile of u_i in the apparent profile of u_i is*

$$\sum_{n=0}^{r} \frac{p_{ij_n}}{r} q_i.$$

The proportion of queries from the real profile of u_i in the apparent profile of $u_m \neq u_i$ is

$$\begin{cases} \frac{p_{ij}}{r} q_i & \text{if } u_m \text{ is collinear with } u_i \\ 0 & \text{otherwise.} \end{cases}$$

Under particular circumstances Proposition 5 has the simpler expression given in Corollary 6.

COROLLARY 6. *Under the same assumptions as in Proposition 5, suppose that all users post queries with the same frequency, so that $q_i = q_j$ for all i, j. Then the proportion of queries from the real profile of u_i in the apparent profile of u_i is $\frac{1}{k}$. The proportion of queries from the real profile of u_i in the apparent profile of another user $u_m \neq u_i$ is $\frac{1}{rk}$ if u_m is collinear with u_i and 0 otherwise.*

Proposition 5 has the following interpretation.

COROLLARY 7. *The users in a community following the P2P UPIR (II) protocol will forward to the server more of the queries posted by themselves than they will forward queries posted by other users.*

This corollary implies that the server can infer the real profile of a user from his apparent profile. The P2P UPIR (II) provides a partial protection of the privacy of the user in front of the server, valid for sparse use. But if we let the protocol run for a while in order to let the user post enough queries to notice the difference, then the users real profile will get inferable from his apparent profile.

We have seen two different strategies for how the user should treat his own queries when implementing P2P UPIR. In the first the user does not forward his own queries to the server and in the second he does. Both provide insufficient privacy protection. Now we will look at a third variation of the P2P UPIR protocol where the user adjust the number of his own queries he should forward to the server so that his real profile results uniformly distributed over the apparent profiles of his neighbors and himself.

The protocol which we call P2P UPIR (III) differs from the P2P UPIR (I) protocol only in the steps the user follows when the decrypted content of the communication space is a query originally posted by himself which is waiting for a user to post it to the server.

The P2P UPIR (III) protocol is obtained from the P2P UPIR (I) protocol by replacing step 2.(c) by:
2.(c) If the outcome is ***a query posted by the user himself****, then u forwards the query to the server with a probability to decide. If u forwards the query to the server, then u also awaits the answer. When u receives the answer, he encrypts it and records it in c. In any case then u restarts the protocol with the intention to post his query;*
The idea behind the modification of the protocol is to adjust the number of his own queries the user forwards to the server in order to obtain a smooth diffusion of his real profile over the apparent profiles of the collection of his neighbors and his own apparent profile.

A finite projective plane is an optimal solution to the problem of preserving the privacy of the user in front of the database, in the sense that it maximizes the number of apparent profiles into which the real profile of a user is diffused, under the restriction to keep the size of the user community fixed [10]. More important, it is the only type of configuration where $N(u)$, the set of users collinear with the user u and different from u, are all the users in the configuration different from u. As already commented, the user who adopts the strategy to not forward any of his own queries (the P2P UPIR (I) protocol) as well as the user who adopts the opposite strategy to forward his own queries (the P2P UPIR (II) protocol) are both hazarding the privacy of their real profiles in front of the server, even when the used configuration is a finite projective plane. The P2P UPIR (III) is an intention to avoid these flaws in privacy by adjusting the number of own queries a user should forward to the server. The idea is to adjust so that a user forwards the correct proportion of his own queries in order for the proportion of his real profile to be constant, or at least asymptotically constant, over the apparent profiles of $\{u\} \cup N(u)$. Adjusting in this way, a finite projective plane indeed does provide privacy for the user, since u's queries are uniformly diffused into the apparent profiles of the users $\{u\} \cup N(u)$, which in a finite projective plane is the whole set of users.

Such an adjustment is possible if the frequencies with which the users post queries is the same for all users, as will be stated in Proposition 8. What is perhaps surprising is that the adjustment is still possible when the frequency with which they post queries is not the same for all users, under the assumption that the users check the communication spaces with equal frequency.

PROPOSITION 8. *Consider a community of users implementing a P2P UPIR protocol with a combinatorial (v, b, r, k)-configuration and impose on the users to check their communication spaces with a fixed frequency higher or equal to the frequency with which they post queries. Then the user u's real profile is optimally diffused into the apparent profiles of $\{u\} \cup N(u)$ if u forwards a proportion of*

$$\frac{1}{r(k-1)+1}$$

of his own queries to the server.

Proposition 8 therefore suggests a change in the protocol so that the users check their communication spaces with a fixed frequency. One should probably choose this frequency to be higher than or equal to the highest frequency with which any user posts queries. In this way, when a user has a query to post, he can post it to the first communication space that he checks. When the user has no query, then he checks the communication space anyway.

4. CONCLUSIONS

We have described some problems regarding the privacy supplied by the P2P UPIR protocol presented in [5, 6]. We have also described how to solve these problems.

5. ACKNOWLEDGMENTS

The authors would like to thank Josep Domingo-Ferrer for many helpful discussions. They would also like to apologize to the AOL users cited in this article, who have not given their approval. The authors think that the citations may be justified by the purpose to improve the privacy for users in the future. The authors were partially supported by the Spanish Government through projects TIN2009-11689 "RIPUP", CONSOLIDER INGENIO 2010 CSD2007-00004 "ARES" and by the Government of Catalonia under grant 2009 SGR 1135. The first author was partially supported by a FPU grant (BOEs 17/11/2009 and 11/10/2010). The authors are with the UNESCO Chair in Data Privacy, but their views do not necessarily reflect those of UNESCO nor commit that organization.

6. REFERENCES

[1] AOL query logs, *www.aolstalker.com*.
[2] Bras-Amorós, M. and Stokes, K. (2009) *The semigroup of combinatorial configurations*, arXiv:0907.4230v3.
[3] Chor, B., Goldreich, O., Kushilevitz, E., Sudan, M. (1998) *Private information retrieval*, Journal of the ACM, 45:965–981.
[4] Domingo-Ferrer, J., González-Nicolás, Ú. (2011) *Rational Behavior in Peer-to-Peer Profile Obfuscation for Anonymous Keyword Search*, Submitted.
[5] Domingo-Ferrer, J., Bras-Amorós, M. (2008) Peer-to-peer private information retrieval, Domingo-Ferrer, J. and Saygin, Y., editors, *Privacy in Statistical Databases, volume 5262 of Lecture Notes in Computer Science*, 315–323. Springer.
[6] Domingo-Ferrer, J., Bras-Amorós, M, Wu, Q., Manjón, J. (2009) *User-private information retrieval based on a peer-to-peer community*, Data Knowl. Eng., 68(11):1237–1252.
[7] Gropp, H. (2007) *Handbook Of Combinatorial Designs (Charles J. Colbourn and Jeffrey H. Dinitz ed.)*, chapter Configurations, Chapman and Hall/CRC, Kenneth H. Rosen, 353–355.
[8] Grünbaum, B.(2009) *Configurations of points and lines*, Graduate Studies in Mathematics, 103, American Mathematical Society.
[9] Lee, J., Stinson, D. R. (2008) *On the construction of practical key predistribution schemes for distributed sensor networks using combinatorial designs*, ACM Trans. Inf. Syst. Secur. 11(2), Article 5, 35 pages.
[10] Stokes, K., Bras-Amorós, M. (2010) *Optimal configurations for peer-to-peer user-private information retrieval*, Computers & Mathematics with Applications, 59(4):1568 - 1577.
[11] Stokes, K., Bras-Amorós, M. (2011) *Associating a numerical semigroup to the triangle-free configurations*, Submitted to Proceedings of Algebraic Combinatorics and Applications (ALCOMA 2010).
[12] Viejo, A., Castellà-Roca, J. (2010) *Using social networks to distort users' profiles generated by web search engines*, Computer Networks, 54(9): 1343 - 1357.