Privacy-Preserving Data Acquisition Protocol

Zbigniew Kwecka, Prof. William J Buchanan, and Duncan Spiers

***Abstract*—Current investigative data acquisition techniques often breach human and natural rights of the data subject and can jeopardize an investigation. Often the investigators need to reveal to the data controller precise details of their suspect's identity or suspect's profile. In this research a novel approach to investigative data acquisition is presented and privacy preserving Investigative Data Acquisition Protocol (IDAP) is defined. This protocol is the first that allows for performing private information retrieval of records matching multiple selection criteria.**

***Index Terms*—cryptography, data acquisition, privacy.**

I. Introduction

Surveys show that the invasion of privacy is among the things people fear the most from the coming years [1][2]. Emerging technologies allow for fast digitalization of operational procedures in many organizations, and depending on how these technologies are used the result can be destructive or beneficial to privacy of the parties involved.

The public authorities such as Police, Customs, and Tax Offices need to request information from third-parties on regular basis and the data protection legislations allow for such requests even without warrants [3][4]. Depending on the way these requests are performed human and natural rights of the data-subject can be breached and/or investigation can be jeopardized [5]. This research gives an insight on how the Privacy Enhancing Technologies (PETs) can be used to improve investigative data acquisition techniques in order to protect the data-subjects and the secrecy of the investigations. It defines a novel, efficient approach to maintain the secrecy of an enquiry.

Motivating application: Police has a profile of a suspect (e.g. sex, age, and ethnic origin) and would like to find individuals fitting this profile working in organizations in a neighborhood to the crime scene, but revealing the profile to these organizations may hurt the investigation and individuals matching the profile.

Currently the police would often have to delay their enquiries in order to protect the investigation as well as the innocent individuals fitting their profile. For example if the case being investigated had a public tension around it, and the suspect's profile was this of a local minority, the enquiry could have serious consequences to the members of this minority.

To generalize the problem the party making the enquiry, the *chooser*, can request k parameters, referred to as return parameters $rp_{1\text{-}k}$ from a *source* table or a view held by the data holder, referred to as the *sender*. While doing so the *chooser* may wish to keep the values $x_{1\text{-}i}$ of the i input parameters $ip_{1\text{-}i}$ secret. Such query can be mapped into an SQL query as follows:

```
SELECT rp1, rp2, …, rpk
FROM source                                   (1)
WHERE ip1=x1 AND ip2=x2  AND … AND ipl=xl
```

This research proposes a novel protocol that could solve the problem at hand using a combination of commutative data locking based on a well known three-pass (3Pass) secret exchange protocol discussed by Shamir in [6][7] and the Private Equijoin (PE) protocol defined in [8].

II. Background And Related Work

Problem of retrieving data records in a private manner from a database hosted by a third-party is not new. Thus, there are a number of primitives that can be used to approach such problem. First there was a Private Information Retrieval (PIR) primitive where *chooser* could query a database in a way that *sender* is unable to identify which row of data is being retrieved. The main motivation behind the PIR schemes is achievement of as low communicational and computational complexity as possible, and the privacy of the records in the database is not a concern [9]. A stronger notion than PIR is *1-out-of-n* Oblivious Transfer (OT) primitive that allows retrieval of a randomly selected record from the dataset of n elements held by *sender*, in a way that *sender* cannot learn which record has been transferred, and *chooser* cannot learn anything about other records in the dataset [10]. *1-out-of-n* OT protocols that allow *chooser* to actively select a record to be retrieved and that have linear or sub-linear complexity are referred to as symmetric PIR protocols. These is the most useful family of the privacy-preserving data retrieval protocols, finding use in electronic watch-lists of suspects

Manuscript received March, 2010. This work was supported in part by the Edinburgh Napier University Research Council.

Z. Kwecka is a Research Student within the Centre for Distributed Computing & Security, Edinburgh Napier University, Edinburgh, EH10 5DT UK (e-mail: z.kwecka@napier.ac.uk).

W. J. Buchanan, is with Centre for Distributed Computing & Security, Edinburgh Napier University, Edinburgh, EH10 5DT, UK (e-mail: z.kwecka@napier.ac.uk).

D. Spierce is with the Centre for Law, Edinburgh Napier University, Edinburgh, EH10 5DT, UK (e-mail: d.spiers@napier.ac.uk).

[11]; cooperative scientific computation [12][13]; and on-line auctions [14].

The information retrieval protocols described above are capable of collecting a record, or records, from a specific index in the *sender's* dataset, but they do not support a search or comparison functionality that could be used to perform an investigation of the dataset. Instead it is often assumed that some indexed description of data is publically available in a form of electronic catalogue [15][16]. Consequently if a system is to allow data retrieval based on private selection, it must implement some private comparison function. There are a number of protocols that allow two parties to compare their values in a private manner, i.e. to compare information without leaking it. But only some are optimized to make comparisons for equality, and these are referred to Private Equality Test (PEqT) protocols. PEqT protocols are often based on commutative [5][11] or homomorphic cryptosystems [15]. Often two different protocols, each with separate and computationally expensive preparation phases need to be used to first make a selection and then retrieve a selected record. The exception to this rule is a range of protocols including: private intersection; private intersection size and PE defined in [8]. For these reason the primitives in [8] are extended in this research to a setting where multiple matches on a given data row must be made in order to retrieve it and unlock it.

Next section describes the building blocks of the investigative data retrieval protocol defined in Section IV. These building blocks are 3Pass primitive based on commutative encryption and the PE protocol proposed in [8]

III. Building Blocks

A. Commutative Cryptosystems

Many cryptographic applications employ sequential encryption and decryption operations under one or more underlying cryptosystems. The reasons to sequence (cascade) different cryptographic schemes together include strengthening the resulting ciphertext and achieving additional functionality impossible under any given encryption scheme on its own [17][18]. A basic cascadable cryptosystem can consist of a number of encryption stages, where output from one stage is treated as an input to another. In such basic cascadable cryptosystem it is necessary to decrypt in the reverse order of encryption operations. However, a special class of sequential cryptosystems, commutative cascadable cryptosystems, allows decryption of a ciphertext in an arbitrary order. Thus, a ciphertext $c = e_b e_a(m)$ (c – ciphertext, m – plaintext, e – encryption operation under keys a and b), could be decrypted as either $m = d_b d_a(c)$ or $m = d_a d_b(c)$. The advantages of such cryptosystems were widely promoted by Shamir in [6] as used in his, Rivest's and Aldman's, now classic, game of *mental poker*, employing the 3Pass secret exchange protocol proposed by them.

This research uses and extends the primitive of the 3Pass protocol in order to lock query results to a set of query variables, as described in Section IV. The 3Pass protocol shown in Fig. 1 was intended so that two parties could share a secret without exchanging any private or public key.

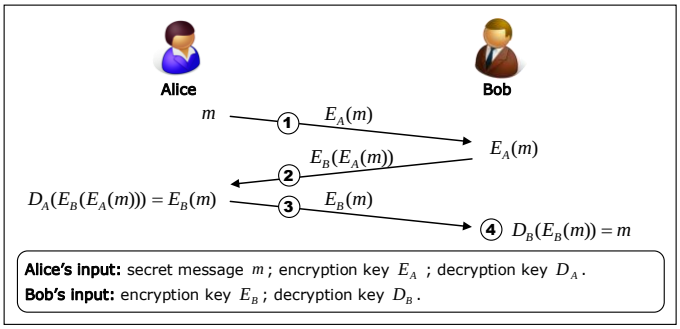

Fig. 1. Three-Pass Secret Exchange Protocol. The protocol was aimed at providing an alternative to public-key encryption and DH-like key negotiation protocols.

The operation of the protocol can be described using the following physical analogy:

1. Alice places a secret message *m* in a box and locks it with a padlock E_A.
2. The box is sent to Bob, who adds his padlock E_B to the latch, and sends the box back to Alice,
3. Alice removes her padlock and passes the box back to Bob,
4. Bob removes his padlock, and this enables him to read the message from inside the box.

There could be more parties, or encryption stages involved in a 3Pass-like protocol. In fact Khayat [19] formalizes this idea into a protocol for sharing a secret with a board of trustees. This protocol itself lacked safeguards needed for a protocol intended for storing information securely for a number of years, since it fails if one of the parties has left the protocol without decrypting the ciphertext. Nevertheless such protocol is ideal for locking a value multiple times and then unlocking it in an arbitrary order, as long as the parties are cooperating until the execution of the protocol is completed. Such functionality is required of the data acquisition protocol described later in this document.

The first known commutative cryptosystem, that could be used to implement the 3Pass protocol, was based on Pohling-Hellman (PH) cryptosystem, asymmetric private key scheme [20]. PH was never a popular protocol, since being asymmetric algorithm it was slow in comparison to other private key encryption algorithms. However, the ever-popular, and still the most widely used Rivest-Shamir-Adleman (RSA) public key cryptosystem is in fact based on the PH algorithm [21], and the RSA cryptographic engine is capable of performing PH encryption and decryption operations. The only difference between these cryptosystems is the fact that operations in PH are performed modulo a single large prime number *p*, whereas RSA has a modulus made up of two large primes. Equations

(2) and (3) show encryption and decryption operations under PH respectively. Encryption exponent/key *e* is selected at random, and the decryption exponent *d* is derived from the modulo-inverse of *e*.

$$c = m^e \bmod p \tag{2}$$

$$m = c^d \bmod p \tag{3}$$

$$e = \log_p c \bmod p \tag{4}$$

In order for the protocol to remain secure both *e* and *d* need to be kept secret. The protocol is commutative for operations modulo a given prime, thus all the parties in the system would have to know *p*. Security is maintained since an adversary with the knowledge of the ciphertext *c* and the prime *p* would need to solve the hard problem (4) in order to determine one of the keys used [10]. However, the PH algorithm shares some weaknesses with it public key successor RSA, and it should only be used to encrypt randomized inputs, such as symmetric encryption keys or hash signatures, and a padding scheme should also be used [22].

When using PH encryption scheme for cascadable commutative cryptographic operations is the fact that looking at the ciphertext, it is impossible to tell how many times it has been locked (encrypted by any party) and whether an operation has worked or not. The ciphertext can be encrypted and decrypted any number of times and there is no way to tell once the plaintext, usually a random number itself, has been reached. Thus, if only two parties are involved in a protocol, and they use one key each, these concerns can be negligible. However, if more parties are engaged in the protocol, such as in the sharing a secret with a board of trustees protocol by Khayat [19] or more than one key is used by any of the parties, such as in the data acquisition protocol proposed in this paper, this can become a concern. This concern has been addressed by Weis in [18] where he describes a method of building a semantically secure commutative cryptosystem from an arbitrary cryptosystem that supports homomorphic multiplication of ciphertexts. Weis provides an example based on ElGamal [EG], homomorphic cryptosystem that uses Diffie-Hellman (DH) key exchange algorithm's foundations. It uses original EG key specification and generation, i.e. first a strong prime *p* and generator *g* of the group Z_p^* are chosen, then a private key *x* is drawn at random, and the public key *y* is calculated from *x*, *g* and *p*. For EG the ciphertext is made of two tuples α and β (5), where *r* is a random number, whereas the commutative EG by Weis (CEG) forms the initial ciphertext in a similar manner, but it is made up of four tuples α, β, γ, and δ. The first two tuples (α, β) form the actual ciphertext, while the other two (γ, δ) are used as a checksum. The calculations involved in creation of the initial ciphertext in CEG, where *r* and *s* are random numbers, are shown in (6). The input to the CEG engine can be a secret message *m*, for the initial encryption of the plaintext *m*, or *n* different 4-tuple ciphertexts in the form $(\alpha,\beta);(\gamma,\delta)$, when the plaintext has already been encrypted *n*-times. In the second case $(n+1)$ values *k* are chosen so that their product is unity, and tuple α of each input ciphertext is multiplied with a distinct value *k*. Finally the sole remaining value *k* is encrypted just like *m* in (6). As the last step of the encryption protocol all *n* input ciphertexts are re-encrypted (7), where $(\alpha',\beta');(\gamma',\delta')$ is the result of re-encrypting $(c^r a, d^r b);(c^s, d^s)$.

$$c = (\alpha,\beta) = (y^r m, g^r) \tag{5}$$

$$c = (\alpha,\beta);(\gamma,\delta) = (y^r m, g^r);(y^s, g^s) \tag{6}$$

$$c = (\alpha',\beta');(\gamma',\delta') = (c^r a, d^r b);(c^s, d^s) \tag{7}$$

The key improvements of CEH over PH is the fact that the ciphertexts contain checksums and a party requested to perform a decryption can identify that the ciphertext received has or has not been previously encrypted by that party. This is done searching the ciphertext for a four-tuple element c_k such that $(\gamma_k / \delta_k^x) = 1$, where *x* is the private key of that party. If such element exists, then the ciphertext has indeed been encrypted by this party, and four-tuple c_k is the element (lock) previously added to the ciphertext by that party.

When choosing whether to apply PH or CEG dissimilarity between the cryptosystems should be noted. Ciphertext formed under PH following a number of encryption operations will be equal to a ciphertext formed using the same encryption operations in an arbitrary order. This property is required by the 3Pass protocol, whereas CEG does not have this property, and thus it cannot be used in the protocol described below or the 3Pass.

B. Private Equijoin Protocol

The most scalable and flexible of PEqT protocols for operations on datasets is the Private Equijoin (PE) protocol proposed in [8] by Agrawal, Evfimievski and Srikant. This scheme extended the classical commutative PEqT shown if Fig. 2, to allow conditional retrieval of data for the records that are common for both parties participating in a *m*-out-of-*n* PEqT protocol.

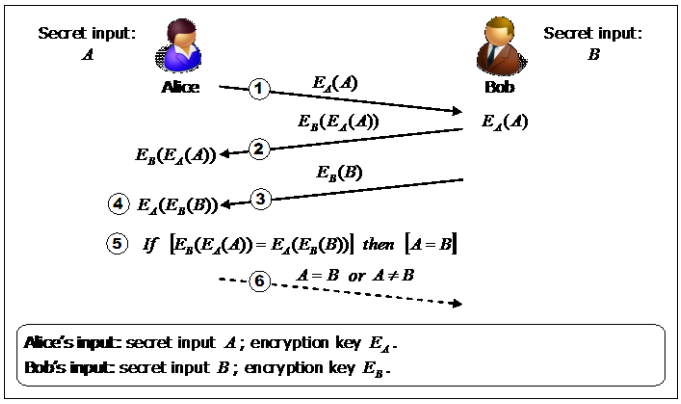

Fig. 2. Private Equality Test. This protocol allows two parties to compare their secret inputs.

The operation of the PEqT protocol shown in Fig. 2 can be described in the following steps:

1. Alice encrypts her input and sends it to Bob,
2. Bob encrypts the ciphertext received from Alice and sends it back,
3. Bob encrypts his secret input and sends it to Alice,
4. Alice encrypts the ciphertext containing Bob's input,
5. Alice compares the two resulting ciphertexts, if they are equal then her and Bob's inputs are equal,
6. Alice may inform Bob about the result.

This simple protocol has been extended in [8] into a PE protocol that enables two parties Alice and Bob to privately compare their sets of unique values V_A and V_B respectively, and allows the requesting party, Alice, to retrieve some extra information about records in V_B that match records in V_A. The PE protocol includes the following steps:

1. Both Alice and Bob apply hash functions *h* to all the elements in their sets, so that $X_A = h(V_S)$ and $X_B = h(V_B)$. Alice chooses a secret PH key E_A at random, and Bob chooses two PH keys E_B and E'_B, all from the same group Z_p^*.
2. Alice encrypts its hashed set: $Y_A = E_A(X_A) = E_A(h(V_S))$
3. Alice sends to Bob her hashed set Y_A, reordered lexicographically.
4. Bob encrypts each entry $y \in Y_A$ with both E_B and E'_B, and for each y sends back to Alice 3-tuple $\langle y, E_B(y), E'_B(y) \rangle$.
5. For each $h(v) \in X_B$, Bob does the following:
   (a) Encrypts $h(v)$ for use in equality test using E_B.
   (b) Encrypts $h(v)$ for use as a key to lock the extra information about *v*, referred to as $ext(v)$, using the E'_B keys: $\kappa(v) = E'_B(h(v))$.
   (c) Encrypts the extra information:
   $c(v) = K(\kappa(v), ext(v))$
   Where *K* is a symmetric encryption function.
   (d) Forms a pair $\langle E_A(h(v)), c(v) \rangle$
   The pairs, containing private match element and the encrypted extra information about record *v*, are then transferred to Alice.
6. Alice removes her encryption E_A from all entries in the 3-tuples received at Step 4 obtaining tuples α, β, and γ such that $\langle \alpha, \beta, \gamma \rangle = \langle h(v), E_B(h(v)), E'_B(h(v)) \rangle$. Thus, α is the hashed value $v \in V_A$, β is the hashed value *v* encrypted using E_B, and γ is the hashed value *v* encrypted using E'_B.
7. Alice sets aside all pairs received in Step 5, whose first entry is one of the β tuples obtained in Step 6. Then using the γ tuples as symmetric keys it decrypts the extra information contained in the second entry in the pair.

Similar results could potentially be achieved when 1-out-of-n PEqT protocol would be combined with 1-out-of-n Oblivious Transfer protocol as demonstrated in [5]. However, as pointed out in Section II mixed solutions usually do not blend well as each requires its own computationally expensive preparation phase.

IV. Investigatory Data Acquisition Protocol

The application scenario and generalization of the problem statement provided in Section I are the motivation behind creation of the Investigatory Data Acquisition Protocol (IDAP). IDAP combines 3Pass protocol and the PE protocols presented in Section III in order to allow private matching against *i* different input parameters $ip_{1\text{-}i}$.

The PE protocol allows conditional retrieval of extra information *ext(v)* for records *v* that are common between *chooser*, and *sender*. Consequently the PE would be a suitable solution for the problem at hand when $i = 1$, but if there is more than one input *ip*, then the following IDAP protocol can used to retrieve the records:

1. *Chooser* provides the *sender* with names of *k* return parameters *rp* and names of *i* input parameters *ip*.
2. Base on the requirements received in Step 1, the *sender* gathers all *rp* and *ip* parameters for all the records into a common SQL view referred to as the *source*, i.e. temporary SQL table. The *chooser* is provided with the schema of this view, describing the data formats used.
3. For each *ip* the *sender* creates a list of unique, or distinct, values. Each such value is associated, for the duration of the enquiry, with a randomly generated commutative key $E_{R(ip_\alpha)}$.
4. *Sender* iterates through the *source* and does the following for each data row *v*:
   (a) Select a random symmetric encryption key $\kappa(v)$.
   (b) Use $\kappa(v)$ to encrypt list of the *k* parameters $rp_{1-k}(v)$
   (c) Encrypt $\kappa(v)$ using *i* commutative keys $E_{R(ip_\alpha)}$ associated with the values of *ip* parameters in row *v*.
   (d) Send to *chooser* pairs $\langle \alpha, \beta \rangle$ made up from:
   α - encrypted $\kappa(v)$ from Step 4(c),
   β - encrypted list of *rp* from Step 4(b)
   , reordered lexicographically.
5. For each ip_α *chooser* engages in a PE protocol to retrieve random commutative keys associated with the interesting value x_α.
6. *Chooser* attempts to decrypt each α in the pairs received in Step 4(d) using all *k* commutative keys retrieved in Step (5).
7. In Step 6 *chooser* obtained key $\kappa(v)$ protecting the return parameters encrypted by *sender* in step 4(b) and

provided to the *chooser* in Step 4(d). Finally *rp* parameters for each data row matching the selection criteria $ip_\alpha = x_\alpha : 1 < \alpha < i$ are obtained.

If a given pair $\langle \alpha, \beta \rangle$ sent to *chooser* in Step 4(d) would not match the selection criteria then depending on the commutative encryption scheme used for locking the key $\kappa(v)$ decryption error would be raised in Step 6 or Step 7. Thus, when PH scheme would be used then no errors would be detected at Stage 6, but the symmetrical decryption process in Step 7 would rise and error, and would not be capable of decrypting the record. In turn if the CEG would be employed then by the use of the checksums it would be discovered in Step 6 that the decryption commutative keys at hand are incapable of decrypting the ciphertext.

Of an interest is the fact that use of CEG in the above protocol would reveal existence of records that match few but not all input parameters. Under CEG the ciphertext is made up from *n* 4-tuples, where *n* is the number of times a given value has been encrypted or locked. Each 4-tuple contains a checksum that can be used in conjunction with the decryption key to verify that a given ciphertext has been locked by a corresponding encryption key. Consequently when CEG is used the *chooser* will know how many records in the *source* match the value x_α for a given input ip_α. Depending on the circumstances this may be beneficial to the enquiry, and acceptable by the data controller.

The above protocol requires a perfect match between the input parameters $x_{1\text{-}i}$ and the respective values in the *source*. Thus, going back to the application scenario from Section I the police could request to obtain data on all female employees that are 26 years old and have a white ethnic origin. However, such a request would not return any results for the records where a data subject is 25 or 27 years old. It is impossible to create a fuzzy match based on the technologies employed, but the police could ask the data controllers to create ranges of values acceptable for a given parameter. In such solution all the records where a data subject would be in their teens, twenties, thirties and so on, could be encrypted using keys common for their age range, and the PE in Step 5 would also use this range name or unique identifier to collect the related commutative key.

V. Evaluation

The communicational complexity of the IDAP is quite small. It is possible to run the protocol with only $O(2+2i)$ data transfers. This property would allow for non-interactive runs of the protocols where the public authorities submit their requests and later return for the results. Consequently, going back to the application scenario businesses that are not yet on-line could use a simple database of their employees or even a spreadsheet as *source* and exchange information with the authorities in person or vie post.

IDAP is computationally expensive, however, being a protocol designed for a specific task of allowing privacy enhanced investigations to take place, the time taken for the protocol to complete a run is negligible in the scale of an enquiry. Complexity of the protocol is dependent of the underlying data, since each unique value for all input parameters needs to be treated separately. Thus, the computational complexity for cases where the input parameters have only 2 possible values will be at its minimum, and when all values of $ip_{1\text{-}i}$ are unique in will be at its maximum. Assuming that the row count of the *source* is *n* the computational complexity of the protocol is shown in Table I.

TABLE I
Computational Complexity of IDAP

		Symmetric Crypto.		Asymmetric Crypto	
		key generation	crypto. operation	key generation	crypto. operation
Step 3	min.	-	-	$O(2i)$	-
	max.	-	-	$O(ni)$	-
Step 4	min.	$O(n)$	$O(n)$	-	$O(ni)$
	max.	$O(n)$	$O(n)$	-	$O(ni)$
Step 5	min.	$O(2i)$	$O(3i)$	$O(3i)$	$O(5i)$
	max.	$O(ni)$	$O(i(n+1))$	$O(3i)$	$O(i(n+3))$
Step 6	min.	-	-	-	$O(ni)$
	max.	-	-	-	$O(ni)$
Step 7	min.	-	$O(n)$	-	-
	max.	-	$O(n)$	-	-
Total	min.	$O(2i+n)$	$O(2n+3i)$	$O(5i)$	$O(2ni+5i)$
	max.	$O(n(i+1))$	$O(2n+ni+i)$	$O(i(n+3))$	$O(3ni+3i)$
Cost ms/operation		0.66	0.33	10	50

The complexity of each of the steps in the IDAP protocol, providing min. and max. value possible for given *n* and *i*. Where *n* is the number of the data rows in the *source* table or view, and *i* is the number of input parameters *ip* in the enquiry. Cost in *ms* for performing given operation from managed C# .NET application is also given.

Table I shows that the total time for the enquiry is linear to both *i* – the number of input parameters *ip*; and *n* – number of the data rows in the *source* table or view. Fig. 3 and 4 illustrate this concept. The max. time required for an investigation with 5 input parameters on 1000 rows of data can be read from Fig. 4 as being just over 800s. These figures were calculated from average cost taken to perform the operations under C# .NET managed code on an ordinary office PC with 1GB of RAM and 1.8GHz CPU. For the problem at hand this is an acceptable result.

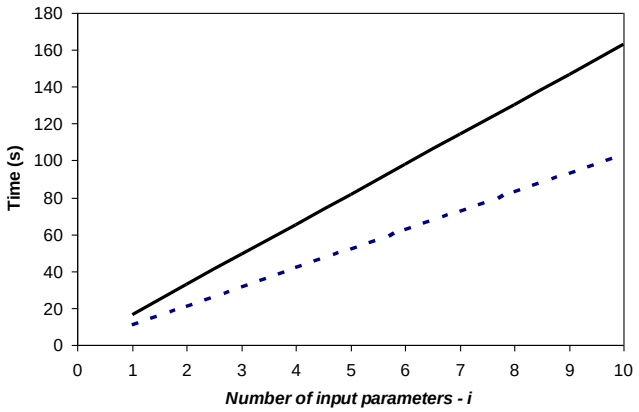

Fig. 3 The max. (and min.) time needed for the IDAP enquiry is linear to the number of input parameters *ip* used as selection criteria. Here the maximum time required is show by the solid line and the minimum is shown by the dashed line. Constant number of data records $n = 100$ was used in this graph.

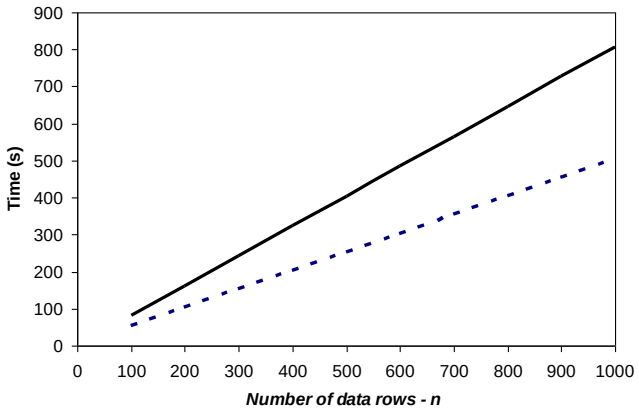

Fig. 4 The max. (and min.) time needed for the IDAP enquiry is linear to the number data rows in the *source* table or view. Here the maximum time required is show by the solid line and the minimum is shown by the dashed line. Constant number of input parameters $i = 5$ was used in this graph.

VI. Conclusion

This paper has shown that creation of privacy-preserving IDAP that allows for more than one private selection criteria in a retrieval protocol is possible. The solution shown is based on the well known secret exchange primitive of 3Pass and the most scalable symmetric PIR protocol found in literature, PE by Agrawal, Evfimievski and Srikant. The complexity of the protocol is relatively high, but taking into consideration that the protocol is designed with real-life investigations, where per case permissions must be granted to retrieve third-party data, time taken per enquiry is negligible. This makes the IDAP protocol valid as a non-intrusive investigation technique for public authorities.

References

[1] P. Swire and L. Steinfeld, "Security and privacy after September 11: the health care example," in *Proceedings of the 12th annual conference on Computers, freedom and privacy*. San Francisco, California: ACM Press, 2002.

[2] D. Balz and C. Deane, "Differing Views on Terrorism," in *The Washington Post*. Washington D.C., 2006, pp. A04.

[3] Crown, "Data Protection Act 1998," TSO, 1998.

[4] EUROPEAN PARLIAMENT, "European Data Protection Directive 95/46/EC," *Official Journal of the European Union*, vol. L, pp. 31-50, 1995.

[5] Z. Kwecka, W. Buchanan, D. Spiers, and L. Saliou, "Validation of 1-N OT Algorithms in Privacy-Preserving Investigations," presented at 7th European Conference on Information Warfare and Security, University of Plymouth, 2008.

[6] A. Shamir, "On the Power of Commutativity in Cryptography," in *Proceedings of the 7th Colloquium on Automata, Languages and Programming*: Springer-Verlag, 1980.

[7] A. Shamir, R. L. Rivest, and L. Adleman, "Mental Poker," in *The Mathematical Gardner*, D. A. Klarner, Ed. Boston, MA: Prindle, Weber & Schmidt, 1981, pp. 37-43.

[8] R. Agrawal, A. Evfimievski, and R. Srikant, "Information sharing across private databases," presented at Proceedings of the 2003 ACM SIGMOD international conference on Management of data, San Diego, California, 2003.

[9] R. Ostrovsky and William E. Skeith III, "A Survey of Single-Database PIR: Techniques and Applications," presented at PKC, 2007.

[10] B. Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*: John Wiley & Sons, Inc., 1995.

[11] K. B. Frikken and M. J. Atallah, "Privacy preserving electronic surveillance," in *Proceedings of the 2003 ACM workshop on Privacy in the electronic society*. Washington, DC: ACM Press, 2003.

[12] W. Du and M. J. Atallah, "Privacy-Preserving Cooperative Scientific Computations," in *Proceedings of the 14th IEEE workshop on Computer Security Foundations*: IEEE Computer Society, 2001.

[13] S. Goldwasser and Y. Lindell, "Secure Computation without Agreement," in *Proceedings of the 16th International Conference on Distributed Computing*: Springer-Verlag, 2002.

[14] C. Cachin, "Efficient private bidding and auctions with an oblivious third party," presented at 6th ACM conference on Computer and communications security - CCS '99, Singapore, 1999.

[15] F. Bao and R. Deng, "Privacy Protection for Transactions of Digital Goods," in *Information and Communications Security*, 2001, pp. 202-213.

[16] B. Aiello, Y. Ishai, and O. Reingold, "Priced Oblivious Transfer: How to Sell Digital Goods," in *Advances in Cryptology — EUROCRYPT 2001*, vol. 2045, *Lecture Notes in Computer Science*, B. Pfitzmann, Ed.: Springer-Verlag, 2001, pp. 119-135.

[17] C. Shannon, "Communication theory of secrecy systems," *Bell System Technical Journal*, vol. 28, 1949.

[18] S. A. Weis, "New Foundations for Efficient Authentication, Commutative Cryptography, and Private Disjointness Testing," in *Department of Electrical Engineering and Computer Science*, vol. PhD. Cambridge, MA: Massachusetts Institute of Technology, 2006, pp. 115.

[19] S. H. Khayat, "Using Commutative Encryption to Share a Secret," 2008.

[20] S. Pohlig and M. Hellman, "An improved algorithm for computing logarithms overGF(p)and its cryptographic significance," *Transactions on Information Theory, IEEE*, vol. 24, pp. 106-110, 1978.

[21] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, pp. 120-126, 1978.

[22] B. Kaliski and R. Laboratories, "RSA Problem" in *ACM SIGKDD Explorations*: MIT Laboratory for Computer Science, 2003, pp. 10.