# SECURITY PROTOCOLS FOR OUTSOURCING DATABASE SERVICES

Tran Khanh DANG

**Abstract:** Advances in networking technologies and the continued growth of the Internet have triggered a new trend towards outsourcing data management and information technology needs to external service providers. As a recent manifestation of this trend, there has been growing interest in outsourcing database services in both the commercial world and the research community. Although the outsourced database service model is emerging as an efficient replacement solution for traditional in-house database management systems, its clients, however, have to store their private data at an external service provider, who is typically not fully trusted, and so it introduces numerous security research challenges. To ensure data confidentiality, the outsourced data is usually encrypted and querying is then carried out with the support of trusted client front-ends or secure coprocessors. Despite a large number of research activities done for securing outsourced databases and removing unencrypted data from exposure to the external server and other intruders, no work has been able to radically secure outsourced databases with associated indexes during the query execution. By exploiting such indexes and with relevant available knowledge, attackers can infer confidential information from the outsourced encrypted data. This article discusses potential attacks in such situations and introduces two security protocols for outsourcing database services. The main contributions focus on solutions to the problem of data privacy/ confidentiality and user privacy. The theoretical analyses show that the proposed protocols can effectively protect outsourced data and its associated indexes as well as the clients against various sophisticated attacks.

**Keywords:** Outsourced Database Services, Data/ User Privacy, Private Information Retrieval/ Storage, Tree-Based Index Structure, Untrusted Server, Encrypted Data.

## Introduction

Advances in networking technologies and the continued growth of the Internet have triggered a new trend towards outsourcing data management and information technology needs to external service providers. As a recent manifestation of this trend, there has been growing interest in outsourcing database services in both the commer-

*INFORMATION & SECURITY. An International Journal, Vol.18, 2006, 85-108.*

cial world and, especially, the research community.[1,2] In the outsourced database service (ODBS) model, clients rely upon external servers and experts for the storage, maintenance, and retrieval of their data. The possibility of outsourcing such database services has generated wide interest in organizations because such a model alleviates their needs to purchase expensive hardware and software, or to pay for professionals to deploy, maintain, and upgrade the system, which are now taken over by the service provider. However, this ODBS model also introduces numerous research challenges and thus has rapidly become one of the most active topics in the research community.[3,4,5,6,7,8,9,10]

As mentioned, in the ODBS model, a client stores its private data at an external service provider who is typically not fully trusted. On the other hand, in this digital age, for most clients, databases take a critical role related directly to their existence and development. Therefore, ensuring clients' data confidentiality is obviously one of the foremost challenges in this model. The question "how is clients' private data protected against sophisticated attackers?" has got much attention from researchers.[11,12] Sophisticated attackers here mean both intruders and insiders, including operators of the external server. Notably, with these malicious insiders, traditional database security techniques [13,14] are useless.

Basically, regardless of the untrusted server at the provider's side, the ultimate goal that clients want is that they can use the outsourced database service as an in-house one. This includes a requirement that clients can operate on their outsourced data without worrying about leak of their sensitive information. This requirement in turn poses several additional challenges related to privacy-preserving for client's queries as well as for the outsourced data during the execution of operations at the untrusted server. Overall, although security requirements are different between real-world applications, the following requirements are most noteworthy:

- *Data confidentiality*: Outsiders and even the server's operators (database administrators) are not able to see the client's outsourced data contents in any case (including when the client's queries are performed on the server).
- *User privacy*: Clients do not want the server to know about their queries and the returned results.
- *Data privacy*: Clients are not allowed to get more information than what they are querying on the server.
- *Authentication and data integrity*: Clients must be ensured that data returned from the untrusted server has originated from the data owner and has not been tampered with.

The above security requirements are different from the traditional database security issues and will in general influence the performance, usability and scalability of the

ODBS model. Among the four, the last security objective (i.e. authentication and data integrity) is out of the scope of this article and we refer interested readers to some recent publications [15,16] for more details. In this article, the author concentrates on addressing the first three security objectives for the outsourced databases that come together with tree-based index structures as discussed below.

To ensure data confidentiality in the ODBS model, outsourced data is usually encrypted before being stored at the external server and querying the data is then carried out with the support of trusted client front-ends [17] or secure coprocessors.[18] This approach can protect the data from outsiders as well as the server, but it introduces difficulties in the querying process. It is hard to protect the user and data privacy as performing queries over encrypted data while still maintaining an acceptable query processing performance. We will elaborate on this issue in the next section with concrete examples.

Although several research activities have been conducted on securing the outsourced database and removing the plaintext (unencrypted data) from exposure to the external server and other intruders,[19,20,21,22,23,24,25] no work has been done to radically secure very large outsourced databases with associated indexes, which are used to accelerate the process of data retrieval. Very large databases augmented by sophisticated and efficient indexes, especially tree-based indexes, are very popular in modern database application domains such as image processing, geographical information systems (GISs), time-series databases, CAD/CAM, and so on.[26] Moreover, not only for such very large databases, the problem of protecting tree-based indexes in traditional RDBMSs and random access files from potential attacks is also important.[27] Basically, the index structures help clients improve the query performance in terms of CPU-, memory-, and IO-cost.[28] By exploiting such (encrypted) indexes and with relevant available knowledge malicious users can infer confidential data/ information from the outsourced *encrypted* data. Some approaches have been recently proposed to deal with this problem.[29] Nevertheless, none of them gives a complete solution to the problem. This article will discuss potential attacks in such situations and introduce two extreme security protocols for outsourcing database services. The proposed novel security protocols employ the state-of-the-art private/ repudiative information retrieval (PIR/RIR) protocols in order to secure both the encrypted data and the associated tree-based indexes to be outsourced against a variety of attacks.

The rest of this article is organized as follows. The next section briefly introduces and discusses related work that has been done or ongoing. Specifically, various approaches to securing the outsourced data will be introduced that resort to both software- and hardware-based solutions, and their weaknesses will be discussed. Next, two new security protocols for outsourced encrypted data with associated tree-based indexes will be introduced. After that, the author discusses and presents possible

changes to these new security protocols in order to balance security and performance. Later, open research issues relevant and indispensable to the real-world application systems are presented. And finally, concluding remarks and future work are given in the last section.

## Related Work and Discussions

Consider the following real-life scenario: An organization *M* has a DNA database containing patterns about various diseases. *M* stores these DNA patterns on a database server *DB* and allows a client *A* to access the database to get information with respect to *A*'s DNA sequence. This scenario poses several security issues as follows:

- If *DB* is an untrusted external server, *M* then has to protect its data contents, i.e. the DNA patterns, from being accessed and analyzed by *DB* and other intruders. This security issue is referred to as data confidentiality in the previous section.
- Whenever *A* accesses *DB*, s/he does not want *M* or even *DB*'s operators to know exactly what she is concerned about, both the query and its result. In other words, *A* is concerned about her privacy (the user privacy issue).
- Client *A* is not allowed to get more information other than what s/he is querying on *DB*. This is an important aspect in the real-world scenarios because *A* may have to pay for what she can get from *DB* and *M* does not allow her to get more than what she has paid for or even *A* does not want to get what she does not need from *DB* and *M* (e.g., because *A* is using a low bandwidth connection, limited memory/ storage devices). This security issue is referred to as data privacy (see the introduction).

The need of data confidentiality, data or user privacy depends on particular scenarios in the ODBS model and this must be considered carefully. For example, if *DB* is hired just for *M* to use, i.e. client *A* is *M* itself and *M* is outsourcing its database services only to make use of the advantages of the ODBS model, then, although the data privacy is unnecessary in this case, neglecting the user privacy as mentioned above may potentially lead to expose the outsourced data to danger, even if they have been encrypted. We will detail this problem later.

In general, protecting outsourced data mainly relates to the three security issues as mentioned above [30] and we now briefly introduce and discuss related work done or ongoing in addressing these issues. Figure 1 below sketches the general service provider models that will be discussed:

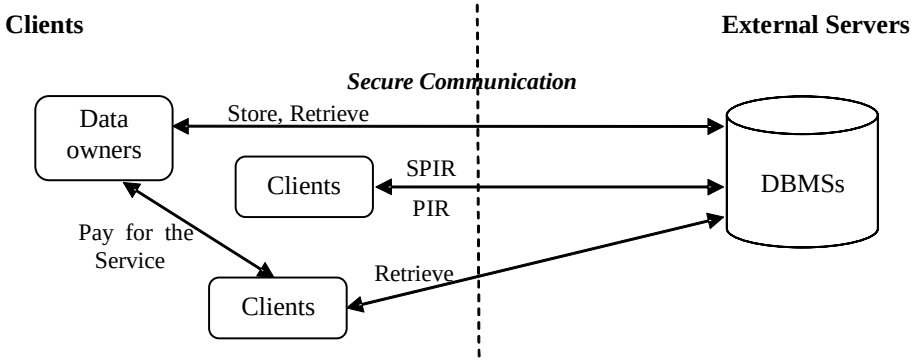


Figure 1: An Overview of Service Provider Models.

As shown in Figure 1, there are four main service provider (SP) models based on the client/ server architecture:[31]

- *UP-DP model*: Data owners are also the SPs. They sell information and charge clients for using their services. The sold information is important and thus the SP is concerned about the data privacy. In this model, the client is concerned about the user privacy. We, therefore, call this outsourcing model the UP-DP model (UP-DP stands for User Privacy – Data Privacy).
- *UP-nDP model*: Similarly to the UP-DP model, data owners here are also the SPs and they charge clients only for using their services, but the stored data is *public*. In this model, the client is also concerned about the user privacy, but the SP is *not* concerned about the data privacy. We, therefore, call this outsourcing model the *UP-nDP* model (nDP stands for not data privacy).
- *DC-UP model*: Data owners are also *unique* clients and their data is outsourced to the external database server. In this model the data owner (also the client) is only concerned about the data confidentiality and the user privacy. We thereafter refer to this outsourcing model as the *DC-UP* model (DC stands for data confidentiality).
- *DC-UP-DP model*: Data owners outsource their data and charge clients for using their data/ information. This is the most complex model in terms of security issues. The data owner is concerned about both the data confidentiality and data privacy with respect to both the external database server and its clients. The client, in turn, is concerned about the user privacy with respect to both the data owner and the server. Moreover, the data owner also takes the client role when accessing its outsourced data on the server and, in this case, the data owner is concerned about the user privacy as well. We, therefore, call this outsourcing model the *DC-UP-DP* model.

We can easily realize that each SP model requires different security objectives and thus different security techniques/ protocols have been invented to satisfy these objectives. In Table 1 we summarize security techniques/ protocols related to the SP models being discussed.

In the UP-DP model, it is not necessary the stored data to be encrypted because the data owner (also the SP) can employ traditional database security techniques to protect their data more efficiently. To satisfy the user privacy requirement, private information retrieval (PIR)-like protocols are employed.

The PIR protocol was first introduced by Chor and colleagues [32] and it has been investigated as well as improved by many researchers thereafter.[33,34] In principle, the PIR protocol allows a client to access a database without revealing to the server both the query and the returned result. More specifically, using the PIR protocol, clients have the possibility of retrieving the *i*-th record of an $N$-record database without revealing the value *i* to the server. In other words, the server does not know what data the client is querying or getting, hence the user privacy is satisfied. In addition, it is easy to observe that security objectives in the UP-nDP model can be solved simply by using any PIR-like protocol.

Notably, Ostrovsky and Shoup [35] have developed the PIR protocol so that it can also support the writing operations privately. Their new protocol is named the private information storage (PIS). Recently, Asonov and Freytag [36] proposed a repudiative information retrieval (RIR) protocol, which is a modified version of the PIR one, to preserve the user privacy but with a better IO-cost for preprocessing before answering a query. The new IO-cost complexity is reduced from $O(N \log N)$ to $O(\sqrt{N})$, where $N$ is the number of records in the database. The main idea of the RIR protocol is the relaxation of the privacy requirement in which some information on the record identity is allowed to be revealed. However, the information revealed should not be enough to indicate definitely if it was record 1, or 2, …, or $N$.

Nevertheless, PIR/ RIR protocols cannot satisfy the data privacy objective, which should also be dealt with in the UP-DP model. Aiming to address this issue, some research work has been carried out. Specifically, Gertner and colleagues have developed a protocol called symmetrically private information retrieval (SPIR) protocol that can be built on the basis of any PIR protocol with the aim to satisfy both user and data privacy requirements.[37] In addition, it should be pointed out here that all approaches developed for the UP-DP model have not been designed to secure tree-structured data against potential attacks. As stated by Du and Atallah, this is not a trivial task and needs much more research.[38] Specifically, whenever applying approaches developed for the UP-DP model where the data are indexed using some tree-based indexing technique, the data privacy will not be satisfied because the com-

parison at a node of the outsourced search tree will give information about the data which is associated with that node. We will consider an example with a B+-tree later.

Table 1: Security Techniques and Protocols Related to the Outsourced Database Service Model.

| ***Security Techniques and Protocols*** | ***Security Objectives*** | | | ***Indexing Support*** | ***References (Can be used for)*** |
|---|---|---|---|---|---|
| | Data Confidentiality | Data Privacy | User Privacy | | |
| PIR/RIR, PIS | | | x | | see Chor, Goldreich, Kushilevitz, and Sudan;[39] Asonov,[40] Chor, Gilboa, and Naor;[41] Ostrovsky and Shoup;[42] Asonov and Freytag [43] (UP-nDP model) |
| SPIR | | x | x | | see Gertner, Ishai, Kushilevitz, and Malkin;[44] and Du and Atallah;[45] data owners also host the server (UP-DP model) |
| Untrusted 3rd parties, Secure coprocessors | x | x | x | | see Smith;[46] Du and Atallah;[47] and Smith and Safford [48] (DC-UP-DP model) |
| Index of range, Hash-based methods | x | | | x | see Damiani, Vimercati, Jajodia, Paraboschi, and Samarati;[49] Hacigümüs, Iyer, Li, and Mehrotra;[50] data owners are also clients (not pay-as-you-use service) |
| User anonymity | | | x | | see the papers by Reiter and Rubin [51,52] (identity hiding) |
| Extreme protocol, Secure coprocessors, Access redundancy and node swapping | x | | x | x | This article and Dang;[53] Smith and Safford;[54] Lin and Candan;[55] and Smith [56] (DC-UP model) |
| Extreme protocol | x | x | x | x | This article (DC-UP-DP model) |

Besides, user privacy in some context also requires user anonymity,[57] which means that not only the user's query and its result are of a concern, but also the user's identity itself needs to be hidden (see Table 1). However, user anonymity solutions still have a lot of limitations in both technical and social aspects.[58] More importantly, even when such user anonymity solutions are employed, the outsourced data is still in danger due to sophisticated attacks as will be discussed below. Furthermore, the UP-DP and UP-nDP models are not of main consideration with respect to the ODBS model. In this article, the DC-UP and DC-UP-DP models are in fact of greater interest.

There are some recent approaches related to the data confidentiality requirement for the ODBS model.[59] Among them, actually, solutions resorting to special hardware equipment have also been investigated and developed.[60] Although these hardware-based solutions may satisfy security objectives in several applications (see Table 1), there are still a matter of controversy.[61,62,63] For the security protocols that will be introduced in this article, it is assumed that such a special hardware is not needed and we rely solely on the available software/ hardware infrastructure. Several recent noteworthy approaches not employing any special hardware were also introduced.[64]

Du and Atallah have introduced protocols for secure remote database access with approximate matching.[65] The problem of answering similarity and approximate queries has been extensively studied by many researchers,[66] but not for outsourced data. The original problem is to search a data repository for some data items that are close to a user's query. The closeness is measured using some metric (e.g., Euclidean metric). Du and Atallah have also proposed solutions to four different e-commerce models, which are quite similar to the presented above four SP models. Their solutions can be used for securing the outsourced data with respect to data confidentiality, data and user privacy where appropriate according to the involved model. Contrary to other related approaches, Aggarwal and coworkers have proposed an approach to outsourcing database services without having to encrypt *all* data fields.[67] This approach needs two non-colluded servers to store the outsourced data and can be used for the DC-UP model. However, all of the above described solutions fail to protect the outsourced data as well as the user privacy in case tree-based index structures are used to access the data more efficiently. Such indexes are an indispensable component to large and high-dimensional databases, which are appearing in many modern database applications as mentioned in a previous section.

Nowadays, there are two approaches aiming to protect the data confidentiality for outsourced indexed data.[68] Both approaches protect the outsourced data from intruders and the server's operators through some encryption method. To process queries over encrypted data, two different solutions have been introduced. Hacigümüs and colleagues have proposed storing, together with the encrypted data, additional indexing information.[69] This information can be used by the untrusted server to select

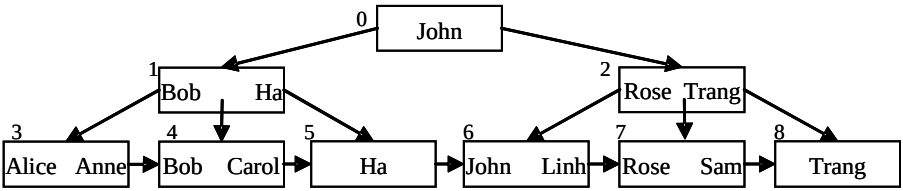


Figure 2: An Example of B+-tree on Attribute *CustomerName*.

the data in response to a user's query. The main idea to process a query in this scheme is to split the original query into: (1) a corresponding query over encrypted relations to run on the untrusted server; and (2) a client query for post-processing the results returned from the server query. The major challenge in this scenario is how to compute and represent index information. Particularly, the relationship between indexes and data should not open the door to inference and linking attacks that can compromise the protection granted by encryption. However, as stated by Damiani and colleagues,[70] although the *index of range* technique proposed by Hacigümüs and team, which relies on partitioning of the domains of client tables' attributes into sets of intervals, is suitable for both exact match and range queries, it introduces difficulties in managing the correspondence between intervals and the actual values present in the database as well as some limitations in such a protection. Similarly to the work of Hacigümüs and colleagues, Damiani and team have also introduced a method to query a tuple-level encrypted database but with a better security level for the outsourced data. For exact match queries, they have analyzed some potential inference and linking attacks and proposed a hash-based indexing method. In order to execute interval-based (range) queries in the ODBS model, they have proposed a solution employing B+-trees typically used in DBMSs.[71] Unfortunately, both of the above approaches do not meet the requirements for user and data privacy (see Table 1). This fact has also been confirmed by Damiani and team. And it can be exploited to carry out inference and linking attacks as will be shown below.

Figure 2 illustrates an example of the B+-tree on an attribute *CustomerName* with sample values. Assume a client/ user is querying all customers whose name is *Ha* on this B+-tree. Following the approach proposed by Damiani and colleagues, the trusted front-end will produce a sequence of queries that will access in sequence nodes 0, 1, and 5. In this case, during the querying process, the user will get more information showing that there are at least two other customers named John and Bob in the database [72] so the data privacy requirement cannot be satisfied. In addition, the server also realizes that the user was accessing nodes 0, 1, and 5, and node 0 is the root, node 1 is an internal node, and node 5 is a leaf node of the tree. Using such information collected gradually, together with statistical methods and data mining tech-

niques, the server can rebuild the whole tree and infer sensitive information from the encrypted database.[73] In this example, this could happen because the user privacy was not protected during the query process.

To protect the user privacy in such cases, there is a recent approach proposed by Lin and Candan.[74] The authors have introduced new techniques to access outsourced tree nodes, called access redundancy and node swapping. The access redundancy technique can be viewed as a computational security version of computational PIR-like protocols, in which information-theoretical security objectives are traded off against performance. Their approach, however, can only be used for the DC-UP model (see Table 1) and has critical limitations, which have been overcome in a recent work conducted by the author.[75] This approach will be discussed in more details in the section on balancing security and efficiency.

As we can see from the analyses above, all introduced approaches that do not employ special security hardware equipment have not dealt radically with potential sophisticated attacks made by exploiting outsourced tree-based index structures. In the next section, general, simple and effective security protocols for securing the outsourced encrypted data with such associated tree-based indexes will be introduced.

## Two Extreme Security Protocols

In this section, we consider two ODBS models as mentioned above: the DC-UP model and the DC-UP-DP model. For these two models, we assume that data of an organization *M* is outsourced to some untrusted external database server *DB*. Moreover, to manage the storage and retrieval of data efficiently, assume that the outsourced data is indexed using tree-based index structures that are the most popular technique and play a fundamental and important role in both traditional and modern database application domains.

*B+Table*

| *NID* | *Node* |
|---|---|
| 0 | (1,John,2,-,-1) |
| 1 | (3,Bob,4,Ha,5) |
| 2 | (6,Rose,7,Trang,8) |
| 3 | (Alice,Anne,4) |
| 4 | (Bob,Carol,5) |
| 5 | (Ha,-,6) |
| 6 | (John,Linh,7) |
| 7 | (Rose,Sam,8) |
| 8 | (Trang,-,-1) |

*B+EncryptedTable*

| *NID* | *EncryptedNode* |
|---|---|
| 0 | D0a1n2g3Kh75nhs& |
| 1 | T9&8ra§ÖÄajh³q91 |
| 2 | H&$uye”µnÜis57ß@ |
| 3 | L?{inh*ß²³&§gnaD |
| 4 | Wh09a/[%?Ö*#Aj2k |
| 5 | j8Hß}[aHo$§angµG |
| 6 | #Xyi29?ß~R@€>Kh |
| 7 | ~B³!jKDÖbd0K3}%§ |
| 8 | T-§µran&gU19=75m |

Figure 3: The Corresponding Plaintext and Encrypted Table Used to Store the B+-Tree at the External Server.

Similarly to other previous approaches, in order to protect the outsourced data from possible intruders we encrypt the data prior to outsourcing. In line with the work of Damiani and colleagues,[76] we choose to encrypt each tree node as a whole since protecting a tree-based index by encrypting each of its fields would disclose to *DB* the ordering relationship between the index values. Moreover, the unit of storage and access in the described approach is also a tree node. Each node is identified by a unique node identifier (NID). The original tree is then stored in *DB* as a table with two attributes: NID and an encrypted value representing the node content. Let us have a look at an example: Figure 3 shows the corresponding plaintext and encrypted table used to store the B+-tree in Figure 2 at the external server. As we can see, the B+-tree is stored at the external server as a table over the schema *B+EncryptedTable = {NID, EncryptedNode}*. A client then retrieves a node from the server by sending a request including the NID of the node.

To ensure the private information storage (PIS) in the future (refer to the previous section), the NID can be assigned arbitrarily by the trusted front-end as a node is inserted. Obviously, to make this feasible, a small amount of meta-data should be kept at the client side. Based on the above settings, in the next two sub-sections general protocols will be succinctly presented in order to meet the security objectives for the two considered ODBS models.

***The DC-UP Model***

In this ODBS model, the data owner is also the unique client so the data privacy objective as mentioned before is not important and could be ignored. As we can observe from the example presented in the previous section (illustrated in Figure 2), even if the data has been encrypted, potential attacks are still possible due to the lack of user privacy-preserving during the querying process. Therefore, in this model, we can simply employ any PIR-like protocol [77] for the client's queries (in this case *M* is also the client) in order to satisfy the user privacy objective. The following formula can be given to ensure the data confidentiality and the user privacy for this model:

$$DC + UP = Encryption + PIR\ protocol \quad (1)$$

Now let us again consider the example from the previous section and the same situation: *M* is querying all customers whose name is *Ha* using the B+-tree as shown in Figure 2 (note that in this model *M* is not concerned about data privacy). Due to the fact that the PIR protocol is employed, the server *DB* does not know which nodes *M* is accessing. The tree information and structure are, therefore, kept secret and no inference and linking attacks are possible.

However, if the data is sometimes changed and *M* needs to update its data on *DB* to reflect the changes, i.e. *M*'s outsourced database is dynamic,[78] we then also need to extend the user privacy requirement so that the server *DB* will not be able to see what have been changed and updated. This is critically important because, with all tree-based index structures, such update operations may lead to node splits.[79] When the split nodes are updated in *DB* and if the server knows this information, which can be collected gradually, it is not difficult to reconstruct the whole tree structure. Then, in this case, the problem of potential inference and linking attacks comes back. To avoid such situations, we need PIS-like protocols in order to protect *M*'s privacy in both reading and writing operations from and to *DB*, respectively. Therefore, we obtain the following formula for this ODBS model:

$$DC + UP = Encryption + PIS\ protocol\ * \quad (2)$$

(* for private reading and writing operations)

The correctness and effectiveness of the proposed protocol could be proved by the theoretical analysis performed in the previous section. Specifically, as demonstrated by Damiani and team,[80] even if the attacker is aware of the distribution of plaintext values in the original database, the outsourced data that has been *encrypted* and *indexed* will still be secure against inference and linking attacks if the index information has been kept secret. In a later section we will further elaborate on the efficiency of this protocol and propose possible changes/ improvements.

***The DC-UP-DP Model***

In this model, assume that *M* is selling its data stored in *DB* and a client *A* is paying for this service. For each query *Q* sent from *A*, both *M* and *DB* should not get any information about *Q* (user privacy) and, in turn, *A* should not get more data/ information from *DB* other than the results of *Q* (data privacy). Note that, in this case, *DB* can even become a client of *M* and it could compromise the privacy of the database by conducting a number of queries and discovering the way the database is encrypted or disguised. A security protocol should defend against this type of active attack. As far as the author is aware, there has been no solution to this model for outsourced tree-structured data. Relying on the solid protocol that has been just proposed for the DC-UP model above, the article proposes a protocol to meet the security requirements of the DC-UP-DP model resorting to a *trusted* third-party, namely *K*. The use of a trusted third-party aims to turn this ODBS model, which is very hard to deal with directly, into the well-behaved DC-UP model.

The assumption for this protocol is that *K* will not collude with *M*, *A*, or *DB* in any way.[81] Furthermore, *K* may send queries to *DB* on behalf of *M* when allowed and up

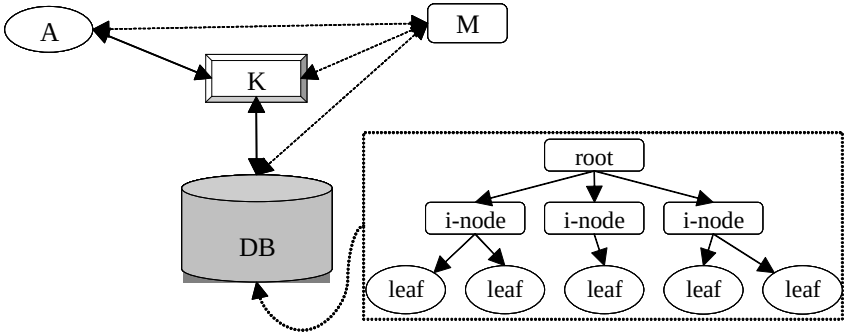


Figure 4: A Security Protocol for the DC-UP-DP Model.

to the level *A* is registered to use *M*'s service s/he can send queries to *K*. It means that *A* can access *M*'s outsourced data indirectly by sending her/his requests to *K*. This protocol is illustrated in Figure 4.

As shown in Figure 4, the outsourced encrypted tree nodes, including the root, the internal nodes (i-nodes), and the leaves, are stored in *DB*. The client *A* pays for the service to the data owner *M* and could query *DB* indirectly via *K*. Thanks to *K*, the third party that both *A* and *M* trust implicitly, the user and data privacy requirements are satisfied. The general steps necessary to perform a query *Q* from *A* are as follows:

1. Client *A* sends a query *Q* to *K*.
2. Upon receiving *Q*, *K* informs *M* (for billing, for example) and waits for approval from *M* in order to access *DB*.
3. Once the information from *K* is received, *M* informs *DB* so that *K* can query *M*'s outsourced database on behalf of *M*. After receiving *DB*'s acknowledgement, *M* informs *K*.
4. From this time on, *K* assumes the role of *M* in the DC-UP model as discussed above, and it accesses *DB* using the security protocol presented in Equation 1 (note that *A* is only able to retrieve information from *DB*, not to update *M*'s outsourced data in *DB*).
5. Finally, *K* filters and returns to *A* only the results of the query *Q*. Obviously, *K* has been informed by *M* what *A* is able to get from the database, but *M* will not be informed what information *A* has got regarding any queries.

In the presented protocol, the trusted third party *K* acts similarly to the secure coprocessors described by some authors.[82,83] This is also the only weakness of the protocol. With the assumption that we could establish such a trusted third party, it is easy to prove that the above protocol ensures all the security objectives for the ODBS model, i.e. data confidentiality, and user and data privacy. Eliminating *K* from this protocol,

while still ensuring all security objectives for the ODBS model, is an open research question and also one of the biggest challenges for future research.

Another point worth mentioning here is that it has been implicitly assumed that the *real* data of the outsourced database is all kept in the tree's leaf nodes. This is, however, not always true with multidimensional access methods (MAMs). With complex data objects to be indexed, the leaf nodes contain only identifiers (IDs) of the data objects, and the data objects are usually kept at a separate place.[84] For such types of index structures, i.e. structures without real data kept in their leaf nodes, we can still apply the same scheme for encryption and storage at the server as follows: (1) the tree nodes are encrypted as a whole and stored in the server as described above (refer to Figure 3), and (2) each real data page that contains the real data objects is encrypted as well, and they will also be stored on the server with additional IDs similarly to the tree nodes. Therefore, both the server and intruders are not able to differentiate between tree nodes and data pages. This uniform storage creates more difficulties in compromising the database confidentiality.

In real-world applications, the extreme security protocols proposed for the two ODBS models above can be modified so as to reduce their communication and computation costs, whereas an acceptable security level is still maintained. In the section below, the author will discuss issues related to the efficiency of the proposed protocols and present some possible modifications.

## Balancing Security and Efficiency

In this section, the efficiency of the proposed security protocols will be discussed. The efficiency in the context here can be interpreted in terms of CPU-, IO- and memory-cost. For many other approaches supporting user privacy,[85] the costs are linear in database size. Obviously, this is an undesirable situation due to the difficulty in deploying such cost-inefficient protocols in real-world applications.

As observed in the section on the DC-UP model, the main factor influencing database access efficiency in the DC-UP model is the efficiency of the employed PIR protocol (note that the PIS protocol that ensures both the private information retrieval and storage is also built on a certain PIR protocol – consider equation (2)). Specifically, as pointed out by Chor and team,[86] the information-theoretic PIR protocol will become prohibitively expensive when only one server is employed to host the outsourced data. This indicates that the proposed in this article protocols (for the two ODBS models) will also become prohibitively expensive if there is no replication of the outsourced data and an information-theoretic PIR protocol is employed. The main question is "*How will the client's queries be performed effectively, efficiently and obliviously over encrypted data without revealing any information about both data*

*and queries to unauthorized people?*" It has motivated the author to look for possible modifications of the protocols in order to make them more practical. Below, such possible modifications for the two protocols introduced for the DC-UP and DC-UP-DP models will be presented.

***Modifications for the DC-UP Model***

As has been already introduced and discussed, the RIR protocol is a modified and improved version of the PIR protocol in terms of cost reduction. In fact, the RIR protocol is a computational PIR protocol, in which the extreme security requirements are relaxed in order to gain a better query performance. Therefore, we can employ the RIR protocol instead of the information-theoretic PIR protocol to reduce the costs for database access. Formula 3 below reflects this change in the presented protocol for the DC-UP model:

$$DC + UP = Encryption + PIR\ protocol \tag{3}$$

Besides, similarly to the PIS protocol, we can also build repudiative information storage (RIS) protocol based on the RIR protocol to support both reading and writing operations privately. The RIS protocol is better than the PIS protocol in terms of IO-cost. Therefore, one can employ the following modified formula for the DC-UP model with *dynamic* outsourced databases and associated tree-based index structures:

$$DC + UP = Encryption + RIS\ protocol\ * \tag{4}$$

(* for repudiative reading and writing operations)

Also, in order to support the oblivious search on a single outsourced search tree (i.e., the replication of the outsourced database as in some PIR/ RIR-like protocols is unnecessary), Lin and Candan present a protocol based on two new techniques: access redundancy and node swapping.[87] With these two techniques and some additional settings, their protocol can be used for the DC-UP model. The two techniques are briefly summarized in what follows.

*Access Redundancy*

This technique requires that whenever a client accesses a node, called target node, she asks for a set of $m$-1 randomly selected nodes in addition to the target node from the server. By this access redundancy, the probability that the server can guess the target node is 1/$m$. Here, $m$ is an adjustable security parameter.

As mentioned, the access redundancy technique can be viewed as a *computational* PIR-like protocol with a better performance, but a worse security level compared with the information-theoretic PIR-like protocols. This technique is also different from the

one presented by Damiani and colleagues,[88] where only the target node is retrieved (this may reveal the tree structure as shown above).

Apart from redundancy in node access, this technique bears also another weakness: it may lead to leak of information about the target node. This could be easily observed: multiple access requests for the root node will reveal its position by simply calculating the intersection of the redundancy sets of the requests. If the root node position is disclosed, there is a high risk that its child nodes (and also the whole tree structure) may also be revealed. This shortcoming could be overcome by secretly changing the target node's address each time it is accessed.

*Node Swapping*

Each time a client requests to access a node from the server, it asks the server for a redundancy set of *m* nodes consisting of at least one *empty* node together with the target one. The client then (1) decrypts the target node; (2) manipulates its data; (3) swaps it with the empty node; and (4) re-encrypts the nodes in the redundancy set and writes them back to the server. As proven by the authors of this technique, with it, the possible position of the target node is randomly distributed over the data storage space on the untrusted server, and thus the weakness of the access redundancy technique is overcome. Note that, in order to prevent the server from differentiating between read and write operations, a read operation is always followed by a write operation for all nodes in the redundancy set back to the server.

Although these techniques are applicable to searching outsourced search trees with sound experimental results reported, it has several limitations and weaknesses. As has been elaborated by the author in a recent publication,[89] this solution can not be applied to dynamic outsourced search trees where data items may be inserted into, removed from, or modified. More importantly, it has also been pointed out that applying this solution directly to such dynamic trees may lead to leak of information about the queries and the tree structure, and so the security objectives are compromised. The author has presented solutions to overcome these limitations and weaknesses as well as to deal with privacy-preserving basic operations (including both search and updates) on outsourced search trees.

***Modifications Related to the DC-UP-DP Model***

First, it is easy to realize that all possible modifications for the DC-UP model can also be *suitably* applied to the DC-UP-DP model (refer to step 4 in the protocol proposed for the DC-UP-DP model). However, it should be noted that the client *A* is only allowed to retrieve the outsourced data but not to update the data. Only the data owner *M* is able to update its outsourced data on the server *DB*. Therefore, all possible modifications presented in the previous subsection can be used for the relevant

operations between *M* and *DB*, while only the protocol as shown in Formula 3 and the access redundancy and node swapping techniques are needed for the operations between the trusted third party *K* and *DB*.

Furthermore, with the DC-UP-DP model, in order to reduce communications complexity, we could store meta-data of the outsourced tree, namely its root and internal nodes, at the trusted third party *K* instead of storing them all on the server *DB*. In this case, there is just a slight change for *DB* compared to the DC-UP model's settings: *DB* now stores only leaf nodes of the tree (and may be real data pages – refer to the section devoted to the DC-UP-DP model). It is not necessary to encrypt the meta-data stored at *K*. Afterwards, in step 4 of the security protocol, *K* first processes *Q* using the meta-data of the database, i.e. the root and leaf nodes of the tree. *K* will access *DB* if it finds it necessary to do so. Similarly, from this time on, *K* takes the role of *M* in the DC-UP model, i.e. *DB* will not know *K*'s queries as well as their results. Note that, in this case *K* will not let *M* know whether it needs to access *DB*. This prevents *M* from inferring that the information that the client *A* needs is currently very likely in its database. Some variants of this approach can also be employed, for example: storing just a part of the tree's meta-data on *K*, but not the root and *all* the internal nodes.

Nevertheless, there is a flaw in the modified scheme just presented: consider the case that *K* has checked its meta-data relevant to the query *Q* and has found that it is unnecessary to access *DB*, and if *K* does so, the data owner *M* is able to discover it. Basically, *M* can carry out this "attack" in various ways. A simple scenario is as follows: *M* colludes with *DB* to see whether *K* will access its outsourced database after *M* has informed *DB* as shown in step 3 of the protocol. As described above, *K* will not access the database if unnecessary and, in that case, both *M* and *DB* will know that what *A* is trying to get does not exist in *M*'s outsourced database (i.e., *Q*'s result set is empty). Therefore, the user privacy is partially not preserved. To resist this kind of attack, *K* will still have to perform some dummy accesses to *DB* even if it has found out that this is no longer necessary to answer *Q*.

## Open Research Questions

Obviously, the first very important question is "Do the proposed security protocols open the door for criminals to carry out fraudulent actions more confidentially?" Computer criminal-related problems have been increasingly growing and now, if we provide clients with the means of hiding their identifiers (e.g., the CROWDS model as introduced in previous sections), their queries, or what they have taken away (the user privacy), how can we protect other clients and organizations (including the service providers) from malicious actions?

The second question, which is somewhat related to the first one, is "How can DBMSs conduct auditing activities in systems provided with such extreme security protocols (without employing special hardware equipment)?" The DBMS may not know who is accessing the system, what they are asking for, and what the system returns to the client, how can it tackle the accountability or develop intrusion detection systems? The goals of privacy-preserving and accountability appear to be in contradiction and an efficient solution to balance the two is still open. More discussions about this topic can be found in a recent publication.[90]

Besides, as already mentioned, avoiding the use of a third party in the DC-UP-DP model is an interesting and challenging problem as well. All of these questions/ problems (and many others) are still open and require future research.

## Conclusions and Future Work

In the ODBS model, the private data is stored at an external service provider, who is typically not fully trusted. Therefore, dealing with security issues in the ODBS model has rapidly become one of the most active topics in the research community. In general, to protect the outsourced data from malicious users, three major issues need to be dealt with radically: data confidentiality (DC), user privacy (UP), and data privacy (DP). For each particular ODBS model, we need to address different security objectives. In this article, the main contributions lie in the following: (1) We have summarized, discussed, and classified different service provider models as well as security techniques and protocols related to them; (2) Two security protocols for the two most popular ODBS models have been introduced, namely the DC-UP and DC-UP-DP models, as well as possible modifications/ improvements have been proposed so that they can scale well to different real-world application domains; and (3) This work has presented important open research directions, which are relevant and necessary for real-world applications.

The proposed protocols for the two ODBS models support outsourced encrypted tree-structured data. This is an important aspect because tree-based index structures have taken a fundamental and crucial role in both traditional and modern database application domains. Especially, the two proposed security protocols have proven to be extreme security protocols for the corresponding ODBS models. They can protect user's data with associated tree-based index structures against various sophisticated attacks from intruders as well as insiders, including the server's operators. To the best of our knowledge, these are among the advanced solutions to the problem of radically securing outsourced data with associated indexes.

Last but not least, considering the fact that the proposed protocols are rather theoretical, the future work will be focused on the open research directions as mentioned, to-

gether with implementing and evaluating the efficiency and effectiveness of these protocols on different real-world application domains. In particular, comparing the efficiency of the protocol for the DC-UP model using some efficient *computational* PIR protocol with the one introduced by the author in another publication will be of particular interest.[91] This will enable the evaluation of the practical value of PIR-like protocols.

**Notes:**

[1] Einar Mykletun, Maithili Narasimha, and Gene Tsudik, "Authentication and Integrity in Outsourced Databases" (paper presented at the 11th Annual Network and Distributed System Security Symposium –NDSS04, California, USA, February 2004).

[2] Tran Khanh Dang, "Privacy-Preserving Basic Operations on Outsourced Search Trees" (paper presented at the International Workshop on Privacy Data Management – PDM05, in conjunction with ICDE05, IEEE Computer Society, Tokyo, Japan, April 2005).

[3] Wenliang Du and Mikhail J. Atallah, "Protocols for Secure Remote Database Access with Approximate Matching" (paper presented at the 7th ACM Conference on Computer and Communications Security, the 1st Workshop on Security and Privacy in E-Commerce, Athens, November 2000).

[4] Sean W. Smith and Dave Safford, "Practical Server Privacy with Secure Coprocessors," *IBM Systems Journal* 40, no. 3 (2001): 683-695.

[5] Hakan Hacigümüs, Bala R. Iyer, and Sharad Mehrotra, "Providing Database as a Service" (paper presented at the 18th International Conference on Data Engineering, San Jose, February-March 2002), 29-40.

[6] Luc Bouganim and Philippe Pucheral, "Chip-Secured Data Access: Confidential Data on Untrusted Servers" (paper presented at the 28th International Conference on Very Large Data Bases, Hong Kong, August 2002), 131-142.

[7] Mykletun, Narasimha, Tsudik, "Authentication and Integrity in Outsourced Databases."

[8] Ping Lin and K. Selçuk Candan, "Hiding Traversal of Tree Structured Data from Untrusted Data Stores" (paper presented at the 2nd International Workshop on Security in Information Systems-WOSIS04, Porto, Portugal, April 2004), 314-323.

[9] Richard Brinkman, Jeroen Doumen, and Willem Jonker, "Using Secret Sharing for Searching in Encrypted Data" (paper presented at the Workshop on Secure Data Management in a Connected World, Toronto, Canada, August 2004), 18-27.

[10] Dang, "Privacy-Preserving Basic Operations on Outsourced Search Trees".

[11] Ernesto Damiani, Sabrina De Capitani di Vimercati, Sushil Jajodia, Stefano Paraboschi, and Pierangela Samarati, "Balancing Confidentiality and Efficiency in Untrusted Relational DBMSs" (paper presented at the 10th ACM Conference on Computer and Communication Security, USA, 27-30 October 2003), 93-102.

[12] Hakan Hacigümüs, Bala R. Iyer, Chen Li, and Sharad Mehrotra, "Executing SQL over Encrypted Data in the Database-Service-Provider Model" (paper presented at the ACM SIGMOD International Conference on Management of Data, USA, June 2002), 216-227.

[13] Silvana Castano, Mariagrazia G. Fugini, Giancarlo Martella, and Pierangela Samarati, *Database Security* (Addison-Wesley and ACM Press, 1994).

[14] Amjad Umar, *Information Security and Auditing in the Digital Age. A Practical and Managerial Perspective* (NGE Solutions, December 2003).

[15] Mykletun, Narasimha, Tsudik, "Authentication and Integrity in Outsourced Databases."

[16] Wenbo Mao, *Modern Cryptography: Theory and Practice* (Prentice Hall PTR, 1st Edition, July 2003).

[17] Damiani, Vimercati, Jajodia, Paraboschi, and Samarati, "Balancing Confidentiality and Efficiency in Untrusted Relational DBMSs."

[18] Smith and Safford, "Practical Server Privacy with Secure Coprocessors."

[19] Yan-Cheng Chang and Michael Mitzenmacher, "Privacy Preserving Keyword Searches on Remote Encrypted Data" (Cryptology ePrint Archive: Report 2004/051), <http://eprint.iacr.org/2004/051> (20 Dec. 2005).

[20] Damiani, Vimercati, Jajodia, Paraboschi, and Samarati, "Balancing Confidentiality and Efficiency."

[21] Hacigümüs, Iyer, Li, and Mehrotra, "Executing SQL over Encrypted Data."

[22] Bouganim and Pucheral, "Chip-Secured Data Access: Confidential Data on Untrusted Servers."

[23] Smith and Safford, "Practical Server Privacy with Secure Coprocessors."

[24] Du and Atallah, "Protocols for Secure Remote Database Access with Approximate Matching."

[25] Dawn Xiaodong Song, David Wagner, and Adrian Perrig, "Practical Techniques for Searches on Encrypted Data" (paper presented at the IEEE Symposium on Security and Privacy, Berkeley, CA, USA, May 2000), 44-55.

[26] Tran Khanh Dang, *Semantic Based Similarity Searches in Database Systems (Multidimensional Access Methods, Similarity Search Algorithms)* (PhD Thesis, FAW-Institute, University of Linz, Austria, May 2003).

[27] Rudolf Bayer and J.K. Metzger, "On the Encipherment of Search Trees and Random Access Files," *ACM Transaction on Database Systems* 1, no. 1 (March 1976): 37-52.

[28] See note 24 for more details about index structures and related algorithms.

[29] Lin and Candan, "Hiding Traversal of Tree Structured Data from Untrusted Data Stores;" Damiani, Vimercati, Jajodia, Paraboschi, and Samarati, "Balancing Confidentiality and Efficiency in Untrusted Relational DBMSs;" Hacigümüs, Iyer, Li, and Mehrotra, "Executing SQL over Encrypted Data in the Database-Service-Provider Model."

[30] We note that the authentication and data integrity issue can be solved independently and separately from these three ones.

[31] Our classification of SP models is quite similar to the one presented in note 3.

[32] Benny Chor, Oded Goldreich, Eyal Kushilevitz, and Madhu Sudan, "Private Information Retrieval" (paper presented at the 36th Annual IEEE Symposium on Foundations of Computer Science, USA, 1995), 41-50.

[33] Dmitri Asonov, "Private Information Retrieval – An Overview and Current Trends" (paper presented at the ECDPvA Workshop, Informatik 2001, Austria, September 2001), 889-894.

[34] Benny Chor, Niv Gilboa, and Moni Naor, "Private Information Retrieval by Keywords," (Technical Report, CS0917, Technion: Israel Institute of Technology, Department of Computer Science, 1997).

[35] Rafail Ostrovsky and Victor Shoup, "Private Information Storage" (paper presented at the 29th ACM Symposium on Theory of Computing, Texas, USA, May 1997), 294-303.

[36] Dmitri Asonov and Johann-Christoph Freytag, "Repudiative Information Retrieval" (paper presented at the ACM Workshop on Privacy in the Electronic Society, Washington, DC, USA, Nov. 2002), 32-40.

[37] Gertner, Ishai, Kushilevitz, and Malkin, "Protecting Data Privacy in Private Information Retrieval Schemes."

[38] Du and Atallah, "Protocols for Secure Remote Database Access with Approximate Matching."

[39] Chor, Goldreich, Kushilevitz, and Sudan, "Private Information Retrieval."

[40] Asonov, "Private Information Retrieval – An Overview and Current Trends."

[41] Chor, Gilboa, and Naor, "Private Information Retrieval by Keywords."

[42] Ostrovsky and Shoup, "Private Information Storage."

[43] Asonov and Freytag, "Repudiative Information Retrieval."

[44] Yael Gertner, Yuval Ishai, Eyal Kushilevitz, and Tal Malkin, "Protecting Data Privacy in Private Information Retrieval Schemes" (paper presented at the 30th Annual ACM Symposium on Theory of Computing, Dallas, Texas, USA, May 1998), 151-160.

[45] Du and Atallah, "Protocols for Secure Remote Database Access with Approximate Matching."

[46] Sean W. Smith, "Secure Coprocessing Applications and Research Issues" (Los Alamos Unclassified Release LA-UR-96-2805, Los Alamos National Laboratory, 1996).

[47] Du and Atallah, "Protocols for Secure Remote Database Access with Approximate Matching."

[48] Smith and Safford, "Practical Server Privacy with Secure Coprocessors."

[49] Damiani, Vimercati, Jajodia, Paraboschi, and Samarati, "Balancing Confidentiality and Efficiency in Untrusted Relational DBMSs."

[50] Hacigümüs, Iyer, Li, and Mehrotra, "Executing SQL over Encrypted Data in the Database-Service-Provider Model."

[51] Michael K. Reiter and Aviel D. Rubin,"Crowds: Anonymity for Web Transactions," *ACM Transactions on Information and System Security* 1, no. 1 (June 1998): 66-92.

[52] Michael K. Reiter and Aviel D. Rubin, "Anonymous Web Transactions with Crowds", *Communications of the ACM* 42, no. 2 (February 1999): 32-38.

[53] Dang, "Privacy-Preserving Basic Operations on Outsourced Search Trees."

[54] Smith and Safford, "Practical Server Privacy with Secure Coprocessors."

[55] Lin and Candan, "Hiding Traversal of Tree Structured Data from Untrusted Data Stores."

[56] Smith, "Secure Coprocessing Applications and Research Issues."

[57] Reiter and Rubin,"Crowds: Anonymity for Web Transactions;" Reiter and Rubin, "Anonymous Web Transactions with Crowds."

[58] Reiter and Rubin, "Anonymous Web Transactions with Crowds."

[59] Du and Atallah, "Protocols for Secure Remote Database Access with Approximate Matching;" Smith and Safford, "Practical Server Privacy with Secure Coprocessors;" Damiani, Vimercati, Jajodia, Paraboschi, and Samarati, "Balancing Confidentiality and Efficiency in Untrusted Relational DBMSs;" Hacigümüs, Iyer, Li, and Mehrotra, "Executing SQL over Encrypted Data in the Database-Service-Provider Model;" Smith, "Secure Coprocessing Applications and Research Issues."

[60] Smith and Safford, "Practical Server Privacy with Secure Coprocessors;" Smith, "Secure Coprocessing Applications and Research Issues".

[61] Oded Goldreich and Rafail Ostrovsky, "Software Protection and Simulation on Oblivious RAMs," *Journal of the ACM* 43, no. 3 (May 1996): 431-473.

[62] Smith, "Secure Coprocessing Applications and Research Issues".

[63] Gagan Aggarwal, Mayank Bawa, Prasanna Ganesan, Hector Garcia-Molina, Krishnaram Kenthapadi, Rajeev Motwani, Utkarsh Srivastava, Dilys Thomas, and Ying Xu, "Two Can Keep a Secret: A Distributed Architecture for Secure Database Services" (paper presented at the 2nd Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 2005), 186-199.

[64] Du and Atallah, "Protocols for Secure Remote Database Access with Approximate Matching;" Damiani, Vimercati, Jajodia, Paraboschi, and Samarati, "Balancing Confidentiality and Efficiency in Untrusted Relational DBMSs;" Hacigümüs, Iyer, Li, and Mehrotra, "Executing SQL over Encrypted Data in the Database-Service-Provider Model;" Aggarwal, Bawa, Ganesan, Garcia-Molina, Kenthapadi, Motwani, Srivastava, Thomas, and Xu, "Two Can Keep a Secret: A Distributed Architecture for Secure Database Services."

[65] Du and Atallah, "Protocols for Secure Remote Database Access with Approximate Matching."

[66] Dang, *Semantic Based Similarity Searches in Database Systems.*

[67] Aggarwal, Bawa, Ganesan, Garcia-Molina, Kenthapadi, Motwani, Srivastava, Thomas, and Xu, "Two Can Keep a Secret: A Distributed Architecture for Secure Database Services."

[68] Damiani, Vimercati, Jajodia, Paraboschi, and Samarati, "Balancing Confidentiality and Efficiency in Untrusted Relational DBMSs;" Hacigümüs, Iyer, Li, and Mehrotra, "Executing SQL over Encrypted Data in the Database-Service-Provider Model."

[69] Hacigümüs, Iyer, Li, and Mehrotra, "Executing SQL over Encrypted Data in the Database-Service-Provider Model."

[70] Damiani, Vimercati, Jajodia, Paraboschi, and Samarati, "Balancing Confidentiality and Efficiency in Untrusted Relational DBMSs."

[71] Note that none of those two approaches supports outsourced multidimensional access methods (MAMs).

[72] The user even gets more information: pointers to nodes 2, 3, and 4, which can be used to access their contents.

[73] See note 11 for more details of possible inference and linking attacks in such situations.

[74] Lin and Candan, "Hiding Traversal of Tree Structured Data from Untrusted Data Stores."

[75] Dang, "Privacy-Preserving Basic Operations on Outsourced Search Trees."

[76] Damiani, Vimercati, Jajodia, Paraboschi, and Samarati, "Balancing Confidentiality and Efficiency in Untrusted Relational DBMSs."

[77] Asonov, "Private Information Retrieval – An Overview and Current Trends."

[78] One disadvantage of the approach introduced by Lin and Candan (see note 8) is that it does not support basic tree operations such as modifications, insertions, and deletions in dynamic outsourced databases.

[79] Dang, *Semantic Based Similarity Searches in Database Systems.*

[80] Damiani, Vimercati, Jajodia, Paraboschi, and Samarati, "Balancing Confidentiality and Efficiency in Untrusted Relational DBMSs."

[81] This is quite similar to an important assumption for PIR-protocols using multiple replication servers: these servers are not colluded with each other. It is also quite similar to the assumption for the secure multi-party computation (SMC) problem, where the parties participating in a computation want to preserve the privacy of their inputs. However, these two problems are essentially different from the problem we are addressing in the DC-UP-DP model. More discussions about the SMC problem can be found in: Wenliang Du and Mikhail J. Atallah, "Secure Multi-Party Computation Problems and Their Applications: A Review and Open Problems" (paper presented at the New Security Paradigms Workshop, USA, September 2001).

[82] We should note that, as introduced by Smith and Safford (see note 4), secure coprocessors cannot be used for the DC-UP-DP model with tree-structured data, but they can only be used for the DC-UP model with or without associated indexes and the DC-UP-DP model without tree-based indexes (see Table 1).

[83] Smith and Safford, "Practical Server Privacy with Secure Coprocessors;" Smith, "Secure Coprocessing Applications and Research Issues."

[84] Dang, *Semantic Based Similarity Searches in Database Systems.*

[85] Du and Atallah, "Protocols for Secure Remote Database Access with Approximate Matching;" Smith and Safford, "Practical Server Privacy with Secure Coprocessors;" Song, Wagner, and Perrig, "Practical Techniques for Searches on Encrypted Data."

[86] Chor, Goldreich, Kushilevitz, and Sudan, "Private Information Retrieval."

[87] Lin and Candan, "Hiding Traversal of Tree Structured Data from Untrusted Data Stores."

[88] Damiani, Vimercati, Jajodia, Paraboschi, and Samarati, "Balancing Confidentiality and Efficiency in Untrusted Relational DBMSs."

[89] Dang, "Privacy-Preserving Basic Operations on Outsourced Search Trees."

[90] Mike Burmester, Yvo Desmedt, Rebecca N. Wright, Alec Yasinsac, "Accountable Privacy" (paper presented at the 12th International Workshop on Security Protocols, Cambridge, UK, 2004).

[91] Dang, "Privacy-Preserving Basic Operations on Outsourced Search Trees."

**Tran Khanh DANG** has been working as a lecturer and researcher in the School of Computing Science, Middlesex University in London (UK) since August 2003. He received his BEng. Degree in Information Technology from the IT Faculty in HCMC University of Technology (Vietnam) in 1998. He achieved the medal awarded for the best graduation student. From 1998 till 2000 he worked as a lecturer and researcher in the IT Faculty. He got a PhD scholarship from the Austrian Exchange Service (OeAD) for the period 2000-2003, and finished his PhD degree (Dr.techn.) in May 2003 at the FAW Institute, Johannes Kepler University of Linz (Austria). Dr. Dang's research interests include database and information security, similarity search and flexible query answering systems, modern information systems and applications, and distributed systems and parallel processing. *Address for Correspondence*: The Burroughs, Hendon, London NW4 4BT, United Kingdom, *E-mail*: k.dang@mdx.ac.uk.