Provably-Secure Schemes for Basic Query Support in Outsourced Databases

Georgios Amanatidis, Alexandra Boldyreva, and Adam O'Neill

Georgia Institute of Technology, USA
amana@math.gatech.edu, {aboldyre,amoneill}@cc.gatech.edu

Abstract. In this paper, we take a closer look at the security of outsourced databases (aka Database-as-the-Service or DAS), a topic of emerging importance. DAS allows users to store sensitive data on a remote, untrusted server and retrieve desired parts of it on request. At first we focus on basic, exact-match query functionality, and then extend our treatment to prefix-matching and, to a more limited extent, range queries as well. We propose several searchable encryption schemes that are not only practical enough for use in DAS in terms of query-processing efficiency but also provably-provide privacy and authenticity of data under new definitions of security that we introduce. The schemes are easy to implement and are based on standard cryptographic primitives such as block ciphers, symmetric encryption schemes, and message authentication codes. As we are some of the first to apply the provable-security framework of modern cryptography to this context, we believe our work will help to properly analyze future schemes and facilitate further research on the subject in general.

1 Introduction

Motivation. Outsourcing data to off-site database service providers is becoming an attractive, cost-effective option for many organizations [40]. In this setting (also known as Database-as-a-Service or DAS), a client stores data on a remote untrusted database server and queries the server in order to receive required portions of the data. Usually this data is stored in the form of a relational database, each divided into records (or tuples) with attributes (or fields). The basic system requirements are (1) query support, (2) computation and communication efficiency for both client and server, and (3) data security. Note that the latter requirement is particularly important in DAS, as data often contains sensitive financial, medical, or intellectual information and the server cannot be trusted. Indeed, ensuring security in DAS is an important research topic that has been receiving increasing attention [25,37,27,26,20,21,37,3,28,2,29,31,9], and security may even be required by law (cf. HIPAA rules [1]).

The problem is that these requirements are in conflict with each other. For example, consider encrypting the data with a secure encryption scheme that hides *all* information and is always randomized (i.e. same messages yields completely different ciphertexts). This does not allow the user to even form a query about

S. Barker, G.-J. Ahn (Eds.): Data and Applications Security 2007, LNCS 4602, pp. 14–30, 2007.

any set of records smaller than the the whole database. Indeed, it turns out that even addressing just the basic exact-match (point) queries is a non-trivial task if one wants to treat security in a systematic, not ad-hoc, way.

Previous Work. Searching on encrypted data has been a topic of multiple relevant works in the cryptographic community, which focus mainly on exact-match queries but in an unsatisfactory way for our context. In particular, the schemes of [41,22,24,15,18,19] provide strong security guarantees (typically revealing only the user access pattern) while allowing a server to answer exact-match queries, but doing so *requires the server to scan the whole database for each query*, yielding unacceptably-slow performance for medium-size to very large databases. The schemes of [19] get around this problem by requiring the (paying) client to know all keywords and all data beforehand and pre-computing a static index for the server that does not allow to treat relational databases. A fundamental question thus becomes what is the best guaranteed security that can be achieved without compromising general efficiency and functionality. The work of [9] recently raised this problem in the asymmetric (public-key) setting, where users explicitly consist of "senders" and "receivers," and provided new security definitions and provably-secure solutions for exact-match queries. We consider this problem entirely in the more-common symmetric-key setting where a client (which may be a large group of users, e.g. in a business) both stores and queries its own data on an untrusted server.

Research on this subject done in the database community focuses on the first two requirements and provides encryption schemes with attractive functionality, namely efficient and optimized indexing and flexible query support e.g. for numerical range, comparison, or aggregation queries [37,3,25,21,27,28,26,31]. In contrast, the security of these schemes is far less clear. Many utilize cryptographic primitives, such as order-preserving hash functions and encryption schemes, which have not been studied by cryptographers, and without scrutinizing their security. For example, using a deterministic encryption scheme for point queries sounds like a reasonable idea, because then forming a point query is feasible and the server can efficiently index and locate the ciphertexts. But what scheme should be used? One common suggestion (see e.g. [28,2]) is to use DES or AES. But these are block ciphers for short plaintexts of at most 128 bits. If a database field holds larger data, say barcode information, then it is not clear how to encrypt longer ones. It would be natural to apply the block cipher block-by-block, but then the adversary will see when the underlying plaintexts have common blocks, which is an unnecessary leak of information. Similarly, fixing the randomness in an arbitrary encryption mode (e.g. CBC) will leak more information than needed.

A noteworthy exception in this body of work is a recent paper by M. Kantarcioglu and C. Clifton [29], which calls for a new direction of research that aims for "efficient encrypted database and query processing with *provable* security properties." Their work provides a first step in this direction. As they observe, unless

one lowers the security bar from the previous cryptographic solutions a linear scan of the database on each query is fundamentally necessary. But the above discussion suggests we must be careful to not go too far. On the other hand, the security definition proposed in [29] requires the use of server-side trusted, tamper-resistant hardware to achieve.

Overview of Contributions. In a broad sense, our goal is to narrow the gap between query-processing-efficient but ad-hoc schemes with unclear security and schemes with strong security guarantees but with unsuitable functionality. We review the provable-security methodology in Section 2. Then to start with, we consider exact-match queries (i.e. with boolean conditions involving only equalities). In Section 4, we formulate what algorithms and properties constitute an *efficiently-searchable authenticated encryption or ESAE* scheme that will allow a server to process such queries, when used to separately encrypt each searchable field, with, unlike for previous cryptographic-community schemes, query-processing efficiency comparable to that for unencrypted databases.

As opposed to previous works in the database community, we go significantly beyond explaining why some attacks do or do not work in order to develop a *foundation* for our understanding of security. Observe that while typically encryption hides all partial information about the data (which is still true for previous searchable schemes in the cryptographic community, and homomorphic encryption schemes in a basic model of security), ESAE cannot because some information needs to be leaked to allow efficient query processing. Hence we formulate a new definition of security that captures the intuition that no adversary should be able to learn *any* useful information about the data within reasonable time, beyond what is unavoidable for the given functionality, namely when two ciphertexts correpond to equal plaintexts; we argue that permitting false-positive results cannot help to hide this correlation in practice. Our definition moreover captures a notion of authenticity that ensures attributes values are not modified or added over the network or at the server side without the user noticing.[1] Thus in a sense we provide the strongest possible notion of security one can reasonably ask for without relying on trusted hardware as in [29]. Note that we do not explicitly model security in the terms of a client-database interaction but always instead simply derive security in this context from that of the "ideal" cryptographic object in question. (This step is crucially absent in [31].) In Section 5 we propose and analyze two exact-match ESAE constructions meeting our definition.

Then in Section 6 we extend our framework to treat prefix-matching queries as well and refer to [4], where we investigate a recent approach [31] to handling range queries and point out some difficulties in achieving a reasonable level of security with it.

[1] The issues of authenticity for the database and the records as a whole, and ensuring that the server returns all the current, requested data, are outside our scope and can be dealt with the methods of [32,35,36,30].

2 The Provable-Security Methodology

Cryptographic protocols were often designed by trial-and-error, where a scheme is implemented and used until some flaws are found and fixed, if possible, and the revised scheme is used until new flaws are found, and so forth. A revolutionary and superior "provable-security" approach was originally proposed by Goldwasser and Micali [23]. The approach requires a formal definition of a security goal (e.g., data privacy) for a given cryptographic object (e.g., an encryption scheme). A security definition comprises a formal description of adversarial capabilities (what an adversary knows and can do) and of what an adversary must do to break the scheme. A proof of security then shows by reduction that a given scheme satisfies the definition under widely accepted assumptions (e.g., that factoring big composite numbers or distinguishing outputs of a block cipher from random strings is hard). The proof thus shows that *the only way* to break the scheme in reasonable time is by breaking the underlying assumption about the hard problem. See [6] for a detailed overview of the provable-security framework.

3 Preliminaries

NOTATION. We refer to members of $\{0,1\}^*$ as strings. If X is a string then $|X|$ denotes its length in bits and if X, Y are strings then $X\|Y$ denotes the concatenation of X and Y. If S is a set then $X \xleftarrow{\$} S$ denotes that X is selected uniformly at random from S. If A is a randomized algorithm then $A(x, y, \ldots; R)$, or $A(x, y, \ldots)$ for short, denotes the result of running A on inputs $x, y, \ldots$ and with coins R, and $a \xleftarrow{\$} A(x, y, \ldots)$ means that we choose R at random and let $a = A(x, y, \ldots; R)$. Oracle access, when given to algorithms (and denoted by superscript), is done as a "black-box," meaning the algorithms see only the input slots provided to them.

SYMMETRIC ENCRYPTION AND MESSAGE AUTHENTICATION. We recall the basics concerning symmetric encryption and, following this, message authentication.

Definition 1. [Symmetric encryption] *A symmetric encryption scheme* $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ *with associated message space* $\mathsf{MsgSp}(\mathcal{SE})$ *consists of three algorithms. (1) The randomized key generation algorithm* $\mathcal{K}$ *returns a secret key* sk*; we write* $sk \xleftarrow{\$} \mathcal{K}$*. (2) The (possibly randomized) encryption algorithm* $\mathcal{E}$ *takes input the secret key* sk *and a plaintext* m *to return a ciphertext; we write* $C \xleftarrow{\$} \mathcal{E}(sk, m)$ *or* $C \leftarrow \mathcal{E}(sk, m; R)$*. If* $C = \mathcal{E}(sk, m, R)$ *for some coins* R *then we say* C *is a* valid *ciphertext for* m *under* sk*. (3) The deterministic decryption algorithm* $\mathcal{D}$ *takes the secret key* sk *and a ciphertext* C *to return the corresponding plaintext or a special symbol* $\perp$ *indicating that the ciphertext was invalid; we write* $m \leftarrow \mathcal{D}(sk, C)$ *(or* $\perp \leftarrow \mathcal{D}(sk, C)$*.)*

Consistency: we require that $\mathcal{D}(sk, (\mathcal{E}(sk, m)) = m$ *for all* $m \in \mathsf{MsgSp}(\mathcal{SE})$*.*

The idea behind security of encryption is that an adversary against a scheme should not be able to deduce anything about the underlying message (except

its length, which encryption cannot hide), upon seeing the ciphertext, even if it has some *a priori* information of its choice about the message. This intuition is captured via a notion of "indistinguishability" of encryptions [11], which requires that no efficient adversary should be able to distinguish between encryptions of two messages, even if the adversary can choose these two messages and request to see ciphertexts of other different messages of its choice.

Definition 2. [**Security of encryption**] *Let $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a symmetric encryption scheme with $\mathsf{MsgSp}(\mathcal{SE})$. Let $\mathcal{LR}$ (left-or-right) be the "selector" that on input m_0, m_1, b returns m_b. The scheme $\mathcal{SE}$ is said to be* secure against chosen-plaintext attack *or* ind-cpa *if for every efficient adversary B the value called the advantage of B $\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE},B}$ is sufficiently small, where*

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE},B} = \Pr[\,\mathbf{Exp}^{\text{ind-cpa-0}}_{\mathcal{SE},B} = 0\,] - \Pr[\,\mathbf{Exp}^{\text{ind-cpa-1}}_{\mathcal{SE},B} = 0\,]$$

and the experiments above are defined for $b \in \{0,1\}$ and an ind-cpa adversary B who is required to query messages of equal length and in $\mathsf{MsgSp}(\mathcal{SE})$, as:

$$\begin{array}{l}\textbf{\textit{Experiment}}\ \mathbf{Exp}^{\text{ind-cpa-}b}_{\mathcal{SE},B} \\ \quad sk \xleftarrow{\$} \mathcal{K}\ ;\ d \xleftarrow{\$} B^{\mathcal{E}(sk,\mathcal{LR}(\cdot,\cdot,b))}\ ;\ \textit{Return } d\end{array}$$

∎

We purposely do not mathematically define an "efficient" adversary and how "small" the advantage should be. This will vary according to the particular appliation. For example, guaranteeing that all adversaries whose running time is up to 2^{60} in some fixed RAM model of computation have maximum advantage 2^{-20} would usually be considered sufficient.

Definition 3. [**MAC**] *A deterministic message authentication code or MAC scheme $\mathcal{MAC} = (\mathcal{K}, \mathcal{M}, \mathcal{V})$ with associated message space $\mathsf{MsgSp}(\mathcal{MAC})$ consists of three algorithms. (1) The randomized key generation algorithm $\mathcal{K}$ returns a a secret key sk; we write $sk \xleftarrow{\$} \mathcal{K}$. (2) The deterministic mac algorithm $\mathcal{M}$ takes input the secret key sk and a plaintext m to return a "mac" for m; we write $\sigma \leftarrow \mathcal{M}(sk, m)$.(3) The deterministic verification algorithm $\mathcal{V}$ takes the secret key sk, a message m, and a mac σ to return a bit $b \in \{0,1\}$; we write $b \leftarrow \mathcal{V}(sk, m, \sigma)$. If b is 1 we say that σ is a valid mac for m under sk.*

Consistency: we require that $\mathcal{V}(sk, m, (\mathcal{M}(sk, m)) = 1$ for all $m \in \mathsf{MsgSp}(\mathcal{MAC})$.

More generally, one can permit $\mathcal{M}$ to flip coins as well, but most practical MACs (e.g., CMAC or HMAC) are deterministic, which is important in our context. Thus in this paper "MAC" means "deterministic MAC."

The standard definition of security of MACs, unforgeability under chosen-message attacks (or uf-cma) requires that no efficient adversary that sees macs of the messages of its choice can produce a valid mac for a new message.

Definition 4. **[Security of MACs]** *A MAC scheme* $\mathcal{MAC} = (\mathcal{K}, \mathcal{M}, \mathcal{V})$ *is said to be unforgeable against chosen-message attack or uf-cma if for every efficient adversary* B *the value* $\mathbf{Adv}^{\text{uf-cma}}_{\mathcal{MAC},B}$ *called* advantage *of* B *is sufficiently small, where*

$$\mathbf{Adv}^{\text{uf-cma}}_{\mathcal{MAC},B} = \Pr[\,\mathbf{Exp}^{\text{uf-cma}}_{\mathcal{MAC},B} = 1\,] \quad \textit{and the experiment is defined as}$$

$$\textbf{\textit{Experiment}}\ \mathbf{Exp}^{\text{uf-cma}}_{\mathcal{MAC},B}$$
$$sk \xleftarrow{\$} \mathcal{K}\,;\ (m,\sigma) \xleftarrow{\$} B^{\mathcal{M}(sk,\cdot),\mathcal{V}(sk,\cdot,\cdot)}\,;\ \textit{Return}\ \mathcal{V}(sk,m,\sigma)$$

and B *is not allowed to query* m *to its mac oracle.* ∎

We will also use an additional property of MACs, namely privacy preservation, originating recently in [12], which requires the outputs of the MAC to hide information about the messages similarly to encryption.

Definition 5. **[Privacy-preserving MACs]** *[7,12] A MAC scheme* $\mathcal{MAC} = (\mathcal{K}, \mathcal{M}, \mathcal{V})$ *is said to be* privacy-preserving *if for every efficient adversary* B *the value called the advantage of* B $\mathbf{Adv}^{\text{pp-mac}}_{\mathcal{MAC},B}$ *is sufficiently small, where*

$$\mathbf{Adv}^{\text{pp-mac}}_{\mathcal{MAC},B} = \Pr[\,\mathbf{Exp}^{\text{pp-mac-0}}_{\mathcal{MAC},B} = 0\,] - \Pr[\,\mathbf{Exp}^{\text{pp-mac-1}}_{\mathcal{MAC},B} = 0\,]$$

and the experiments above are defined for the adversary B *and* $, b \in \{0,1\}$ *as*

$$\textbf{\textit{Experiment}}\ \mathbf{Exp}^{\text{pp-mac-}b}_{\mathcal{MAC},B}$$
$$sk \xleftarrow{\$} \mathcal{K}\,;\ d \xleftarrow{\$} B^{\mathcal{M}(sk,\mathcal{LR}(\cdot,\cdot,b))}\,;\ \textit{Return}\ d$$

Above $\mathcal{LR}$ *is the oracle that on input* m_0, m_1, b *returns* m_b*; and we require that for any sequence of oracle queries* $(m_{1,1}, m_{1,2}), \ldots, (m_{q,1}, m_{q,2})$ *that* B *can make to its oracle, there does not exist any* $m_{i,1} = m_{j,1}$ *or* $m_{i,2} = m_{j,2}$ *for* $i \neq j$ *and moreover* $|m_{i,1}| = |m_{i,2}|$ *for all* i*.* ∎

4 Efficiently-Searchable Authenticated Encryption

What is ESAE. We now define the syntax of an ESAE (Efficiently-Searchable Authenticated Encryption) scheme.

Definition 6. **[ESAE]** *Let* $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ *be a symmetric encryption scheme. We say that* $\mathcal{ESAE} = (\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathcal{F}, \mathcal{G})$ *an* efficiently-searchable authenticated encryption *(*ESAE*) scheme if* $\mathcal{K}, \mathcal{E}, \mathcal{D}$ *are the algorithms of a regular encryption scheme and* $\mathcal{F}, \mathcal{G}$*, are deterministic efficient algorithms where the former takes a secret key and message as input and the latter takes a ciphertext and:*
(1) Completeness:

$$\Pr\left[\, sk \xleftarrow{\$} \mathcal{K}\,;\ f_1 \leftarrow \mathcal{F}(sk, m_1)\,;\ g_1 \leftarrow \mathcal{G}(\mathcal{E}(sk, m_1)) \ :\ f_1 = g_1 \,\right] = 1 \quad \textit{and}$$

(2) Soundness:

$$\Pr\left[\, sk \xleftarrow{\$} \mathcal{K}\,;\ (m_0, m_1) \xleftarrow{\$} \mathcal{M}_{\mathcal{SE}} \ :\ \mathcal{F}(sk, m_0) = \mathcal{G}(\mathcal{E}(sk, m_1)) \,\right] \textit{is sufficiently small}$$

for every message $m_1 \in \mathsf{MsgSp}(\mathcal{SE})$ *and every efficient randomized algorithm* $\mathcal{M}_{\mathcal{SE}}$ *that outputs distinct messages* $m_0, m_1 \in \mathsf{MsgSp}(\mathcal{SE})$. *We refer to the output of* $\mathcal{F}, \mathcal{G}$ *as the* tag *of a message* m *or a corresponding ciphertext* C.

The algorithm $\mathcal{F}$ is used by the user to form queries, and $\mathcal{G}$ is needed by the server to be able to index the encrypted data *a priori*, using the standard data structures (e.g. B-tress), and locate records on request (see below), for which it is crucial that $\mathcal{F}, \mathcal{G}$ are not randomized. Thus the completeness property ensures that encrypted data can be efficiently searched, in logarithmic-time in the database size, meaning this time has not gone up over unencrypted data. The soundness property ensures that false positives do not occur too often so that post-processing is efficient. We first focus on the case that the soundness probability in the definition so small that each ciphertext essentially has a unique tag; we will address increasing the number false-positive results later.

Note that exact-match functionality can also be used to build various other useful more-complicated query types. These include equijoin and group-by, the latter of which is especially useful for example in supporting multi-faceted search that projects among various dimensions (e.g. features/types of products). Moreover, the server can *ipso facto* compute counts over the data, which would also be useful in this context for example to support a product search interface that shows there are, say, 100 CRT and 200 LCD monitors in the database, and 100 15", 100 17", and 100 20" monitors. You click on LCD monitors link and it now shows 50 15", 75 17", and 75 20" such monitors.

Security of ESAE. Efficient "searchability" (ensured by the completeness property) necessarily violates the standard ind-cpa security for encryption. Thus we provide a relaxed definition suitable for given functionality. Completeness implies that the server (and the adversary) will always be able to see what ciphertexts correspond to equal plaintexts, and a security definition should ensure that this is *all* the adversary can learn. To this end we design an indistinguishability experiment (cf. Definition 2) where we disallow the adversary from seeing ciphertexts of equal messages such that it can trivially succeed. The adversary can also mount chosen-ciphertext attacks according to a relaxed chosen-ciphertext-security definition [5,16] that is suitable for our application. For integrity of the data, we also want to require that it is hard produce a new ciphertext or change the existing one without the user noticing, which corresponds to a notion of ciphertext-integrity for authenticated encryption [13].

Definition 7. [Security of ESAE] *Let* $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathcal{F}, \mathcal{G})$ *be an ESAE scheme. Let* $\mathcal{LR}$ *(left-or-right) be the selector that on input* m_0, m_1, b *returns* m_b. *Let* B *be an adversary who is given access to two oracles (called lr-encryption and the decryption oracles). For* $b \in \{0,1\}$ *define the experiment:*

Experiment $\mathbf{Exp}^{\text{ind-esae-}b}_{\mathcal{SE},B}$
$sk \xleftarrow{\$} \mathcal{K}$; $d \xleftarrow{\$} B^{\mathcal{E}(sk,\mathcal{LR}(\cdot,\cdot,b)),\mathcal{D}(sk,\cdot)}$
If $m \neq \perp$ *was returned from* $\mathcal{D}(sk,\cdot)$ *at any point then* $d \leftarrow b$
Return d

We call B an esae adversary *if for any sequence of queries $(m_{1,1}, m_{1,2}), \ldots, (m_{q,1}, m_{q,2})$ that B can make to its lr-encryption oracle, there does not exist any $m_{i,1} = m_{j,1}$ or $m_{k,2} = m_{l,2}$ for $i \neq j, k \neq l$ such that $m_{i,2} \neq m_{j,2}$ or $m_{k,1} \neq m_{l,1}$, in addition to the usual requirements that $|m_{i,1}| = |m_{i,2}|$ for all i and if B does not query the decryption oracle on a ciphertext that has the same tag as any ciphertext that has been returned by the lr-encryption oracle. The* advantage *of an esae adversary B is defined as follows:*

$$\mathbf{Adv}^{\text{ind-esae}}_{\mathcal{SE},B} = \Pr[\,\mathbf{Exp}^{\text{ind-esae-0}}_{\mathcal{SE},B} = 0\,] - \Pr[\,\mathbf{Exp}^{\text{ind-esae-1}}_{\mathcal{SE},B} = 0\,].$$

The ESAE scheme $\mathcal{SE}$ is said to be esae-secure *if for every efficient esae adversary B the function $\mathbf{Adv}^{\text{ind-esae}}_{\mathcal{SE},B}$ is sufficiently small.* ∎

We note the similarity of ESAE to deterministic authenticated encryption (DAE), studied in [39] in the context of transporting (encrypted) symmetric keys. However, the definition of security for DAE in [39] is shown there be equivalent to that for "pseudorandom injections," and we will see that an ESAE scheme need not be pseudorandom nor deterministic.

Discussion. In the context of DAS, the server receives queries with tags for the data, the former of which it would have computed itself, thus the definition of security we provide essentially guarantees that the server cannot learn anything about the data of the user beyond its occurrence profile (or distribution), i.e. how many times a given attribute value (without knowing anything else about it) occurs in the database and in which records, even if it is one of only two possible such values that it can pick itself, and analogously the user access pattern.

As for authenticity (aka. integrity) of ciphertexts, our definition guarantees integrity in that any modification or substitution (malicious or not) to the encrypted data is detected by the user. We note that authenticity is ensured at the field level, and not on the record level or for the entire database; an adversary can still, for example, switch (encrypted) attribute values stored in different records. If the data is updated and returned as whole records, then one can simply authenticate at the record level instead. In many applications, the server can be trusted to return the correct ciphertexts to its paying customers (even when it may try to learn and sell their data). Thus one should mainly protect against non-adversarial transmission or storage errors, and our definition does it.

Increased False-Positives. It seems intuitive that permitting false positive results (i.e. relaxing the soundness condition in Definition 6) via a "bucketization" technique where a fixed number of randomly-chosen plaintexts correspond to each tag, [34,33,17], though requiring the client to do more work to filter out these false-positives, would allow a proportional increase in security by preventing the adversary from correlating equal plaintexts. But we claim that this intuition is not always correct; in practice such information may still be leaked. To see this, consider the *a posteriori* probability of a plaintext occurring a certain number of times given an occurrence distribution on the buckets; the "farther" the latter is from the uniform distribution means a better estimate on the

plaintext occurrence profile, and one cannot expect anything close to the uniform distribution in practice. One solution would be make the bucket distribution instead depend on that of the input, but in particular as noted in [34] this would require impractical communication cost between client and server as this distrbution changes over time, and it is noted in [31] that such mappings are typically not efficiently computable, making storing and managing them impractical.

Comparison to the model from [29] . The security definition of [29] guarantees that an adversary (e.g. the server) cannot distinguish between two queries whose results sets have the same size, whereas ours reveals which records are accessed by such queries. This hold even with respect to extremely powerful adversaries who can mount chosen-ciphertext attacks, whereas our definition applies to somewhat more passive adversaries, which we nevertheless believe is reasonable for the given application. On the other hand, the definition of [29] requires server-side trusted hardware to achieve.

5 Proposed Constructions and Their Security Analyses

Mac-and-encrypt. We first present an easy-to-implement, "off the shelf" way to construct an ESAE scheme from any encryption and MAC schemes and then analyze its security and comment on implementation.

Definition 8. [Mac-and-encrypt construction] *Let $\mathcal{SE} = (\mathcal{K}_E, \mathcal{E}, \mathcal{D})$ be a symmetric encryption scheme and $\mathcal{MAC} = (\mathcal{K}_M, \mathcal{M}, \mathcal{V})$ be a message authentication code. Then we define a new symmetric encryption scheme $\mathcal{SE}^* = (\mathcal{K}^*, \mathcal{E}^*, \mathcal{D}^*, \mathcal{F}, \mathcal{G})$, whose constituent algorithms work as follows:*

- *$\mathcal{K}^*$ sets $sk_M \xleftarrow{\$} \mathcal{K}_M$ and $sk_E \xleftarrow{\$} \mathcal{K}_E$, then outputs $sk_M \| sk_E$.*
- *$\mathcal{E}^*$ on input $sk_M \| sk_E, m$, sets $\sigma \leftarrow \mathcal{M}(sk_M, m)$ and $C \xleftarrow{\$} \mathcal{E}(sk_E, m)$, then outputs $\sigma \| C$.*
- *$\mathcal{D}^*$ on input $sk_M \| sk_E, \sigma \| C$, first sets $m \leftarrow \mathcal{D}(sk_E, C)$ and then $b \leftarrow \mathcal{V}(sk_M, m, \sigma)$. It outputs m if $b = 1$ and $\perp$ otherwise.*
- *$\mathcal{F}$ and $\mathcal{G}$ on inputs $sk_M \| sk_E, m$ and $\sigma \| C$, respectively, return $\mathcal{M}(sk_M, m)$ and σ.*

We first argue that $\mathcal{SE}^*$ is an ESAE scheme if $\mathcal{MAC}$ is uf-cma. Clearly completeness is satisfied. The soundness condition relies on the uf-cma security of $\mathcal{MAC}$. Namely, suppose $\mathcal{MAC}$ is uf-cma but there is an algorithm $\mathcal{M}_{\mathcal{SE}}$ that outputs m_0, m_1 such that $\mathcal{M}(sk_M, m_0) = \mathcal{M}(sk_M, m_1)$ with high probability. This violates uf-cma security as follows. We construct a uf-cma adversary B as per Definition 4 that first runs $\mathcal{M}_{\mathcal{SE}}$ to receive its output (m_0, m_1) then queries its signing oracle for $\mathcal{M}(sk, m_0)$ to get back σ, and finally itself returns (m_1, σ). By the forgoing assumption on $\mathcal{M}_{\mathcal{SE}}$ this adversary has high uf-cma advantage, a contradiction.

Theorem 1. *Let $\mathcal{SE} = (\mathcal{K}_E, \mathcal{E}, \mathcal{D})$ be a symmetric encryption scheme and $\mathcal{MAC} = (\mathcal{K}_M, \mathcal{M}, \mathcal{V})$ be a deterministic MAC. Then let $\mathcal{SE}^* = (\mathcal{K}^*, \mathcal{E}^*, \mathcal{D}^*, \mathcal{F}, \mathcal{G})$ be the*

mac-and-encrypt ESAE scheme defined according to Definition 8. We have that $\mathcal{SE}^*$ is esae-secure if $\mathcal{SE}$ is ind-cpa and $\mathcal{MAC}$ is uf-cma and privacy-preserving.

Due to lack of space the concrete security statement that shows explicit relations between the advantages of the adversaries and the proof are given in [4].

There are many efficient and standardized provably-secure symmetric encryption and MAC schemes that can be used to build an ESAE scheme according to Definition 8. Our recommendations for encryption schemes include CBC and CTR (aka the counter or XOR) encryption modes based on the AES block cipher, which are proven to be ind-cpa under the assumption that AES is a pseudorandom function (PRF) [11]. For MACs, one can use SHA-1 or SHA-256 and AES-based HMAC or CMAC (a variation of CBC-MAC), proven uf-cma assuming the underlying hash function is collision-resistant or PRF and the block cipher is PRF [10,7,14]. Theorem 1 implies that the resulting mac-and-encrypt ESAE is secure under the respective assumptions.

We remark that in database literature (e.g. [25]), some proposed solutions for this problem suggest to use a "random one-to-one mapping" whose output is included with a ciphertext, in order to facilitate "searchability." Thus one interesting implication of the above result is that such a map need not be random, or even pseudorandom, in order to achieve the best-possible notion of security.

Encrypt-with-mac. We now present a construction that is more computation-efficient on the client side and more communication-efficient over the network. This can be crucial, for example, when users have a low-bandwidth connection to the database or are connecting via a battery-constrained device [35]. The idea is to use the mac of the plaintext "inside" the encryption, namely as the randomness used in the encryption algorithm of a standard encryption scheme.

Definition 9. [Encrypt-with-mac construction] *Let $\mathcal{SE} = (\mathcal{K}_E, \mathcal{E}, \mathcal{D})$ be a symmetric encryption scheme and $\mathcal{MAC} = (\mathcal{K}_M, \mathcal{M}, \mathcal{V})$ be a deterministic MAC. Then we define a new symmetric encryption scheme $\mathcal{SE}^* = (\mathcal{K}^*, \mathcal{E}^*, \mathcal{D}^*, \mathcal{F}, \mathcal{G})$, whose constituent algorithms work as follows:*

- *$\mathcal{K}^*$ sets $sk_M \xleftarrow{\$} \mathcal{K}_M$ and $sk_E \xleftarrow{\$} \mathcal{K}_E$, then outputs $sk_M \| sk_E$.*
- *$\mathcal{E}^*$ on input $sk_M \| sk_E, m$, sets $\sigma \leftarrow \mathcal{M}(sk_M, m)$ and $C \leftarrow \mathcal{E}(sk_E, m; \sigma)$, then outputs C.*
- *$\mathcal{D}^*$ on input $sk_M \| sk_E, C$, first sets $m \leftarrow \mathcal{D}(sk_E, C)$. It outputs m if $C = \mathcal{E}(sk_E, m; \mathcal{M}(sk_M, m))$ and $\perp$ otherwise.*
- *$\mathcal{F}$ is same as $\mathcal{E}^*$. $\mathcal{G}$ on input C returns C.*

To see that $\mathcal{SE}^*$ is an ESAE scheme, we note that the completeness requirement is clearly satisfied and the probability in the soundness requirement is zero here due to the consistency requirement in Definition 1.

Ideally, we would like to prove that the above construction is ease-secure assuming that $\mathcal{MAC}$ is a uf-cma and $\mathcal{SE}^*$ is ind-cpa secure. However, slightly stronger assumptions turns out to be needed, but they are met by practical schemes anyway. First, we will need the mac algorithm of $\mathcal{MAC}$ to be a

pseudorandom function (PRF). Naturally, this requires a mac to "look like random bits" without the secret key, a well-studied notion formalized as follows.

Definition 10. *A family of functions is a map* $F\colon \{0,1\}^b \times \{0,1\}^c \to \{0,1\}^c$*, where we regard* $\{0,1\}^b$ *as the* keyspace *for the function family in that a* key $k \in \{0,1\}^b$ *induces a particular function from this family, which we denote by* $F(k,\cdot)$*. The family* F *is said to be* pseudorandom *(or a PRF) if for every efficient adversary* B *given oracle access to a function, its prf-advantage*

$$\mathbf{Adv}^{\mathrm{prf}}_{F,B} = Pr\left[B^{F(k,\cdot)} = 0\right] - \Pr\left[B^{Q(\cdot)} = 0\right]$$

is sufficiently small, where $F(k,\cdot)$ *is the oracle for a random instance of* F *(specified by a randomly chosen key* k*) and* $Q(\cdot)$ *is the oracle for a truly random function with the domain and range of* $F(k,\cdot)$*. Pseudorandom permutations (PRPs) are defined analogously, and in this case the adversary* B *above is also given an inversion oracle.*

To define the assumption needed for encryption, let us say that an encryption scheme $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ has a *max-collision probability* [9] $mc_{\mathcal{SE}}$ if we have that:

$$\Pr\left[\mathcal{E}(sk, m, R_1) = \mathcal{E}(sk, m, R_2)\right] \leq mc_{\mathcal{SE}} \ ,$$

for every $m \in \mathsf{MsgSp}(\mathcal{SE})$, where the probability is taken over the random choices of the key sk and coins R_1, R_2 (chosen independently).

All practical encryption schemes satisfy the above property. The proof of the following is in [4]. It also contains the concrete security statement. .

Theorem 2. *Let* $\mathcal{SE} = (\mathcal{K}_E, \mathcal{E}, \mathcal{D})$ *be a symmetric encryption scheme and* $\mathcal{MAC} = (\mathcal{K}_M, \mathcal{M}, \mathcal{V})$ *be a deterministic MAC. Let* $\mathcal{SE}^* = (\mathcal{K}^*, \mathcal{E}^*, \mathcal{D}^*)$ *be the encrypt-with-mac ESAE scheme defined via Definition 9. Then* $\mathcal{SE}^*$ *is esae-secure if* $\mathcal{MAC}$ *is a PRF and* $\mathcal{SE}$ *is ind-cpa and has sufficiently small max-collision probability.*

The same recommendations for the underlying schemes (CBC, CTR modes, and HMAC and CMAC) we gave for the mac-and-encrypt construct apply here. As we mentioned, CBC and CTR are proven to be ind-cpa assuming the base block cipher is PRF. Randomized CBC and CTR have max-collision probability 2^{-128} when used with AES and the counter-based CTR has zero max-collision probability. HMAC was recently proved to be a PRF assuming the underlying hash function is PRF [7], and CMAC is known to be PRF assuming the base block cipher is PRF; Theorem 2 implies that the resulting encrypt-with-mac ESAE scheme is secure under these respective assumptions.

We remark that our construction is similar to the SIV ("synthetic initialization vector") construction for deterministic authenticated encryption (DAE) in [39]. Indeed, it is straightforward to check that a secure DAE scheme as defined in [39] is also secure as an ESAE scheme. However, our construction and analysis is in fact somewhat more general than the SIV construction, which pertains only to some "initialization-vector-based" symmetric encryption schemes (including CBC and CTR) that implicitly guarantee to meet the max-collision requirement we pinpoint for security.

6 Prefix-Preserving ESAE

Prefix-Matching Queries. We extend our ESAE framework to encryption that allows to efficiently process prefix-matching queries, i.e. locating records whose attribute value starts with a given prefix, for example all phone numbers starting with area-code 310.

Our treatment builds on the study of "online ciphers" (so-called because they can be used on streaming data without buffering) in [8], which we view here as deterministic length-preserving encryption schemes whose input is composed of fixed-length blocks (which we call "characters" of the prefixes), where the ith block of the output depends only on the first i blocks of the input. Thus if two plaintexts agree on their first k characters then so do their ciphertexts. Following Definition 4, to show this implies efficient prefix-searchability (via appropriate server-side index structures for the tuples) we make functions $\mathcal{F}, \mathcal{G}$ return the encryption of an l-character prefix and the first l characters of a ciphertext; the fact that completeness is one and soundness is zero follows from the fact that the encryption is deterministic.

In our construction, the characters of a prefix will be of the input-length for an underlying block cipher (e.g. 64 bits or 4 UTF-16 characters using DES-variants). At the cost of revealing more information to the server for a more flexible granularity of prefixes in the queries ,a *bitwise* prefix-preserving scheme of Xu et al. [42] can similarly be used here (an issue we will return to later), which makes one block cipher computation per *bit* of the input. However, that this may be too inefficient for, say, text files as input. Moreover, as for our previous schemes our construction also achieves ciphertext-integrity, whereas it seems hard to somehow modify the former to achieve such a notion.[2]

Security. The stronger security definition for an online cipher in [8] requires it to be indistinguishable from an "ideal" object that is a function drawn at random from a family of all possible such "online" permutations with the corresponding domain, even when given access to the corresponding "inverter" decryption oracle. Note that for example applying encryption character-by-character is completely insecure: encryptions of "HAT" and "BAT" should look totally unrelated in this setting despite sharing a suffix. We also formulate an additional property of ciphertext-integrity, and thus the encryption algorithm should contain some redundancy at the end so the ciphertext is verifiable. For our definition, we use an ideal object that encrypts a message with a random block appended, and the decryption oracle in the ideal experiment always returns $\perp$ to capture the intuition that the adversary should not be able to create a new valid ciphertext. The novelty of our definition is its generality: it uses only the ideal object in question and without any specific redundancy.

Definition 11. [Security of prefix-preserving ESAE] *Let $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a length- and prefix-preserving symmetric encryption scheme whose message*

[2] Of course, one can always achieve authenticity using a MAC on top of the encryption scheme, but the point is that this would be excessive in some applications.

space $\mathsf{MsgSp}(\mathcal{SE})$ *contains messages of multiple of block-length* n *and let* d *be the maximum possible number of blocks (hereafter we denote the set of such strings by* $D_{d,n}$*). Let* $\mathsf{OPerm}_{d,n}$ *denote the family of all length- and prefix-preserving permutations on* $D_{d,n}$*. Let* $\perp(\cdot)$ *denote the oracle that always returns* $\perp$ *and* r *denote a random* n*-bit block (picked fresh each time it is encountered). For an adversary* A *with access to two oracles define the experiments:*

Experiment $\mathbf{Exp}^{\text{pp-0}}_{\mathcal{SE},A}$	***Experiment*** $\mathbf{Exp}^{\text{pp-1}}_{\mathcal{SE},A}$
$sk \xleftarrow{\$} \mathcal{K}$; $d \xleftarrow{\$} A^{\mathcal{E}(sk,\cdot),\mathcal{D}(sk,\cdot)}$	$g \xleftarrow{\$} \mathsf{OPerm}_{d+1,n}$; $d \xleftarrow{\$} A^{g(\cdot\|\|r),\perp(\cdot)}$
Return d	*Return* d

We call A *a* pp-adversary *if it never repeats queries, never queries a response from its first oracle to its second, and all queries to its first oracle belong to* $D_{d,n}$ *and queries to its second belong to* $D_{d+1,n}$*. The advantage of a* A *is defined as*

$$\mathbf{Adv}^{\text{pp}}_{\mathcal{SE},A} = \Pr[\,\mathbf{Exp}^{\text{pp-0}}_{\mathcal{SE},A} = 0\,] - \Pr[\,\mathbf{Exp}^{\text{pp-1}}_{\mathcal{SE},A} = 0\,].$$

The scheme $\mathcal{SE}$ *is said to be* pp-secure *if for every efficient pp-adversary* A *the probability* $\mathbf{Adv}^{\text{pp}}_{\mathcal{SE},B}$ *is sufficiently small.* ∎

Discussion. Analogous to the case of exact-match queries, our security definition here ensures that the server cannot learn anything about the data except which attribute values share a same prefix, which is obviously unavoidable in this context, where the granularity of such prefix-correlation is given by the length of the block cipher used in our construction below (and on the other hand it is bit-wise for the less-efficient, no-authenticity scheme of [42]). Here one has to be wary of frequency-based (in terms how many *distinct* plaintexts with a given prefix occur in the database) deduction of some prefixes when using text data, which may require adding bogus data to balance these frequencies. We stress that this analysis holds *only* in the presence of prefix-matching (or exact-match) queries. In a generalization and refinement of the approach of [31] that we present in [4], we show that our scheme can in some sense be used to efficiently support range-queries as well, but the security analysis is more delicate.

Our Construction and Analysis. As in [8], appealing constructions such as the authenticated encryption scheme OCB [38] with fixed IV can be shown insecure under Definition 11. We design a prefix-preserving ESAE scheme based on an interesting modification of the HPCBC cipher [8, Construction 8.1] that appends an all-zero block to a message to encrypt and uses a different block cipher on this last block to also achieve ciphertext-integrity, which may also be of independent interest.[3] It is efficient and uses one block cipher and one hash function operation per block of input.

[3] In fact our construction treats HPCPC as a black-box so any on-line cipher that is OPRP-CCA (see [8] for the definition) can be used, but we suggest HPCBC for concreteness.

Definition 12. **[HCBC+]** *Let* $E\colon \{0,1\}^{ek} \times \{0,1\}^n \to \{0,1\}^n$ *be a block cipher. Let* $H\colon \{0,1\}^{hk} \times \{0,1\}^{2n} \to \{0,1\}^n$ *be a family of functions. We associate to them a prefix-preserving ESAE scheme* $\mathsf{HPCBC}^+ = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ *defined as follows. The key generation algorithm chooses randomly a key* $eK \| eK' \| hK$ *where* eK, eK' *are (independent) keys for* E *and* hK *is a key for* H*. The encryption and decryption algorithms are defined as follows:*

Algorithm $\mathcal{E}(eK \| eK' \| hK, m)$	***Algorithm*** $\mathcal{D}(eK \| eK' \| hK, C)$
{ *Parse* m *as* $m[1] \dots m[l]$	{*Parse* C *as* $C[1] \dots C[l+1]$ *with* $l \geq 1$
$C[0] \leftarrow 0^n$; $m[0] \leftarrow 0^n$	$C[0] \leftarrow 0^n$; $m[0] \leftarrow 0^n$
For $i = 1, \dots, l$ *do*	*For* $i = 1, \dots, l$ *do*
$R \leftarrow m[i-1] \| C[i-1]$	$R \leftarrow m[i-1] \| C[i-1]$
$P[i] \leftarrow H(hK, R) \oplus m[i]$	$P[i] \leftarrow E^{-1}(eK, C[i] \oplus H(hK, R))$
$C[i] \leftarrow E(eK, P[i]) \oplus H(hK, R)$}	$m[i] \leftarrow H(hK, R) \oplus P[i]$}
$R \leftarrow m[l] \| C[l]$	$R \leftarrow m[l] \| C[l]$
$P[l+1] \leftarrow H(hK, R) \oplus 0^n$	$P[l+1] \leftarrow E^{-1}(eK', C[l+1] \oplus H(hK, R))$
$C[l+1] \leftarrow E(eK', P[l+1]) \oplus H(hK, R)$	$m[l+1] \leftarrow H(hK, R) \oplus P[l+1]$
Return $C[1] \dots C[l+1]$	*If* $m[l+1] = 0^n$ *then return* $m[1] \dots m[l+1]$
	Else return $\perp$

We note that the 6 first lines of the algorithms (i.e. the part between braces) could be expressed more compactly as $C[1] \dots C[l] \leftarrow \mathsf{HPCBC}(eK \| hK, m)$ and $m[1] \dots m[l] \leftarrow \mathsf{HPCBC}^{-1}(eK \| hK, C)$. This explicit description of HPCBC is given here for completeness. To see the benefit of using our construction over plain HPCBC note that encryption along with a separate MAC (e.g. CMAC) to additionally achieve integrity would roughly double the computation time, making two passes over the input, as compared to our construction.

Security of the scheme is based on the security of the underlying block cipher and the hash function. The corresponding definitions of PRP-CCA security of a block cipher and of almost-xor-universal hash functions is recalled in [8]. AES is believed to be PRP-CCA, and [8] provide references for secure hash function constructions. The proof of the following theorem is in [4]. It also contains the concrete security statement.

Theorem 3. *Let* $E\colon \{0,1\}^{ek} \times \{0,1\}^n \to \{0,1\}^n$ *be a block cipher that is a PRP-CCA. and let* $H\colon \{0,1\}^{hk} \times \{0,1\}^{2n} \to \{0,1\}^n$ *be an almost-xor-universal family of hash functions. Then* HPCBC^+ *defined via Definition 12 is a pp-secure prefix-preserving ESAE scheme.*

6.1 On Efficient Range-Query Processing

In [31] it is shown that encrypting data via a bit-wise prefix-preserving scheme allows efficient (as opposed to scanning the whole database) range queries over the data by specifying the possible prefixes for a desired range. Introducing our prefix-preserving ESAE as well provides a generalized approach, where the block size is not just one bit but a variable parameter. It is shown in [31] that certain attacks are possible if their scheme is used for range queries. In the full version of the paper [4], we generalize such attacks and discuss what is the best level of security prefix-preserving schemes can provide in this context.

Acknowledgments

We thank Brian Cooper and Andrey Balmin for useful comments and references.

References

1. The final HIPAA security rule. Federal Register (2003) Available at http://www.hipaadvisory.com/regs/finalsecurity/index.htm,
2. Aggarwal, G., Bawa, M., Ganesan, P., Garcia-Molina, H., Kenthapadi, K., Motwani, R., Srivastava, U., Thomas, D., Xu, Y.: Two can keep a secret: A distributed architecture for secure database services. In: CIDR 2005
3. Agrawal, R., Kiernan, J., Srikant, R., Xu, Y.: Order preserving encryption for numeric data. In: SIGMOD 2004
4. Amanatidis, G., Boldyreva, A., O'Neill, A.: New security models and provably-secure schemes for basic query support in outsourced databases. A full version of this paper (2007) Available at www-static.cc.gatech.edu/~aboldyre/publications.html
5. An, J.-H., Dodis, Y., Rabin, T.: On the security of joint signature and encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, Springer, Heidelberg (2002)
6. Bellare, M.: Practice-oriented provable-security. In: Information Security Workshop, ISW (1997)
7. Bellare, M.: New proofs for NMAC and HMAC: Security without collision-resistance. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, Springer, Heidelberg (2006)
8. Bellare, M., Boldyreva, A., Knudsen, L.R., Namprempre, C.: Online ciphers and the Hash-CBC construction. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, Springer, Heidelberg (2001)
9. Bellare, M., Boldyreva, A., O'Neill, A.: Efficiently-searchable and deterministic asymmetric encryption. Cryptology ePrint Archive, Report, /186, 2006. (2006), http://eprint.iacr.org/2006/186/
10. Bellare, M., Canetti, R., Krawczyk, H.: Keying hash functions for message authentication. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, Springer, Heidelberg (1996)
11. Bellare, M., Desai, A., Jokipii, E., Rogaway, P.: A concrete security treatment of symmetric encryption. In: FOCS (1997)
12. Bellare, M., Kohno, T., Namprempre, C.: Breaking and provably repairing the SSH authenticated encryption scheme: A case study of the Encode-then-Encrypt-and-MAC paradigm. In: ACM Transactions on Information and System Security. vol. 7(2) (2004)
13. Bellare, M., Namprempre, C.: Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, Springer, Heidelberg (2000)
14. Black, J., Rogaway, P.: CBC MACs for arbitrary-length messages: The three-key constructions. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, Springer, Heidelberg (2000)
15. Boneh, D., Di Crescenzo, G., Ostrovsky, R., Persiano, G.: Public key encryption with keyword search. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, Springer, Heidelberg (2004)

16. Canetti, R., Krawczyk, H., Nielsen, J.: Relaxing chosen-ciphertext security. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, Springer, Heidelberg (2003)
17. Ceselli, A., Damiani, E., De Capitani, d.S., Jajodia, S., Paraboschi, S., Samarati, P.: Modeling and assessing inference exposure in encrypted databases. ACM Trans. Inf. Syst. Secur. 8(1), 119–152 (2005)
18. Chang, Y.-C., Mitzenmacher, M.: Privacy preserving keyword searches on remote encrypted data. In: Ioannidis, J., Keromytis, A.D., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, Springer, Heidelberg (2005)
19. Curtmola, R., Garay, J., Kamara, S., Ostrovsky, R.: Searchable symmetric encryption: Improved definitions and efficient constructions. Cryptology ePrint Archive, Report 2006/210 (2006)
20. Damiani, E., De Capitani di Vimercati, S., Paraboschi, S., Samarati, P.: Computing range queries on obfuscated data. In: Information Processing and Management of Uncertainty in Knowledge-Based Systems (2004)
21. Damiani, E., De Capitani Vimercati, S., Jajodia, S., Paraboschi, S., Samarati, P.: Balancing confidentiality and efficiency in untrusted relational DBMSs. In: CCS (2003)
22. Goh, E.-J.: Secure indexes. Cryptology ePrint Archive, Report 2003/216 (2003) `http://eprint.iacr.org/2003/216/`.
23. Goldwasser, S., Micali, S.: Probabilistic encryption. In: Journal of Computer and Systems Sciencies, vol. 28 (1984)
24. Golle, P., Staddon, J., Waters, B.: Secure conjunctive keyword search over encrypted data. In: Applied Cryptography and Network Security Conference
25. Hacigümüs, H., Iyer, B., Li, C., Mehrotra, S.: Executing SQL over encrypted data in the database-service-provider model. In: SIGMOD (2002)
26. Hacigümüs, H., Iyer, B.R., Mehrotra, S.: Efficient execution of aggregation queries over encrypted relational databases. In: Lee, Y., Li, J., Whang, K.-Y., Lee, D. (eds.) DASFAA 2004. LNCS, vol. 2973, Springer, Heidelberg (2004)
27. Hore, B., Mehrotra, S., Tsudik, G.: A privacy-preserving index for range queries. In: VLDB (2004)
28. Iyer, B.R., Mehrotra, S., Mykletun, E., Tsudik, G., Wu, Y.: A framework for efficient storage security in RDBMS. In: EDBT (2004)
29. Kantracioglu, M., Clifton, C.: Security issues in querying encrypted data. In: DBSec (2005)
30. Li, F., Hadjieleftheriou, M., Kollios, G., Reyzin, L.: Dynamic authenticated index structures for outsourced databases. In: SIGMOD, ACM Press, New York (2006)
31. Li, J., Omiecinski, E.: Efficiency and security trade-off in supporting range queries on encrypted databases. In: DBSec (2005)
32. Mykletun, E., Narasimha, M., Tsudik, G.: Authentication and integrity in outsourced databases. In: NDSS (2004)
33. Mykletun, E., Tsudik, G.: Incorporating a secure coprocessor in the database-as-a-service model. In: International Workshop on Innovative Architecture for Future Generation High Performance Processors and Systems (2005)
34. Mykletun, E., Tsudik, G.: Aggregation queries in the database-as-a-service model. In: DBSEC (2006)
35. Narasimha, M., Tsudik, G.: DSAC: integrity for outsourced databases with signature aggregation and chaining. In: CIKM (2005)
36. Narasimha, M., Tsudik, G.: Authentication of outsourced databases using signature aggregation and chaining. In: Lee, M.L., Tan, K.-L., Wuwongse, V. (eds.) DASFAA 2006. LNCS, vol. 3882, Springer, Heidelberg (2006)

37. Özsoyoglu, G., Singer, D.A., Chung, S.S.: Anti-tamper databases: Querying encrypted databases. In: DBSec, pp. 133–146 (2003)
38. Rogaway, P., Bellare, M., Black, J., Krovetz, T.: OCB: a block-cipher mode of operation for efficient authenticated encryption. In: ACM CCS (2001)
39. Rogaway, P., Shrimpton, T.: A provable-security treatment of the key-wrap problem. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, Springer, Heidelberg (2006)
40. Arsenal Digital Solutions. Top 10 reasons to outsource remote data protection. `http://www.arsenaldigital.com/services/remote_data_protection.htm`
41. Song, D.X., Wagner, D., Perrig, A.: Practical techniques for searches on encrypted data. In: IEEE Symposium on Security and Privacy (2000)
42. Xu, J., Fan, J., Ammar, M.H., Moon, S.B.: Prefix-preserving IP address anonymization: Measurement-based security evaluation and a new cryptography-based scheme. In: ICNP (2002)