Deterministic and Efficiently Searchable Encryption

Michael Wang

March 12th, 2012

Goals of the Paper

To provide some background to the paper, in the database literature, the naive solution of providing public key encryption on untrusted database servers (for example, data stored in a rented cloud server) includes encrypting every record non-deterministically in order to provide notions of privacy in the database. In other words, in order to find particular records in an encrypted database, the entire database would have to be scanned (in other words, searches would take time linear to the size of the database), as indexes cannot be built on non-deterministic ciphertext values. In this paper, the authors detail systems where database systems can maintain provable security bounds via encryption, while allowing logarithmic search bounds that are desired of database systems. They achieve this via deterministic public key encryption, which they prove in the paper that while normally deterministic public key encryptions do not grant the level of security expected in all applications, the deterministic public key encryption algorithms provided in the paper still maintain a degree of provable security against chosen plaintext attacks. The authors, in the last section of the paper, also include extensions to cover chosen ciphertext attacks.

Relevant New Security Definition

The author's definition of privacy, shortened to PRIV in the paper, encompass two notably weaker definitions than the standard definitions for randomized settings. The first deals with the issue that privacy is impossible when the domain of the plaintext space (in other words, the domain of the records in the database) is small. Thus, privacy is only required when the plaintext is drawn from a space of large min-entropy. The second condition for privacy as used in the paper deals with the issue that the ciphertext may be indicative of the plaintext. In short, this second condition deals with partial information that might be available. The authors deal with this by only requiring non-leakage of partial information when the plaintext and partial information do not depend on the public key. The authors reason that this second condition is reasonable for practical reasons, due to the public key used to access the databases are usually hidden within the database software, thus not being known the user, and also that the data will not depend on the public keys.

In implementing this new privacy construct, the authors present two encryption functions that have provable security advantages presented in the paper and extend these two encryption functions to allowing log-time searchable encryption. These two encryption functions are "Encrypt with Hash", and RSA-DOAEP.

Encrypt-with-Hash

In defining this algorithm, the authors use the following definitions of variables:

- Let AE $= (\mathcal{K}, \mathcal{E}, \mathcal{D})$ by any public key encryption scheme.
- Say that $\mathcal{E}(1^k, pk, x)$ draws its coins from a set $\mathbf{Coins}_{pk}(|x|)$

- We write $\mathcal{E}(1^k, pk, x; R)$ for the output of $\mathcal{E}$ on inputs pk, x, and coins R.
- Let $H : \{0,1\}^* \rightarrow \{0,1\}^*$ be a hash function with the property that $H(pk||x) \in \mathbf{Coins}_{pk}(|x|)$ for all $pk, x \in \{0,1\}^*$.

To spare the reader details of the algorithm, to sum all of these definitions and their use, the encryption process calls $H(pk||x)$, passes the result into the encryption scheme as a deterministic encoding of the coins used in the encryption scheme, and outputs the result as the ciphertext. On the decryption, a ciphertext y is decrypted into x, the decryption algorithm regenerates the hash value for the coins, and re-encrypts x. If x is re-encrypted into y, x is returned, otherwise, $\perp$ is returned.

With regards to this algorithm, suppose there is a privacy adversary $A = (A_m, A_g)$ against EwH (encrypt with hash) with min-entropy μ, which outputs vectors of size v with components of length n and makes at most q_h queries to its hash oracle. The resultant advantage function for an IND-CPA adversary B against AE, whose details in the proof are recorded in the appendix, comes out as:

$$\mathbf{Adv}^{priv}_{EwH,A} \leq \mathbf{Adv}^{ind-cpa}_{AE,B} + \frac{2q_h v}{2^\mu} + 8q_h v \cdot mpk_{AE} \tag{1}$$

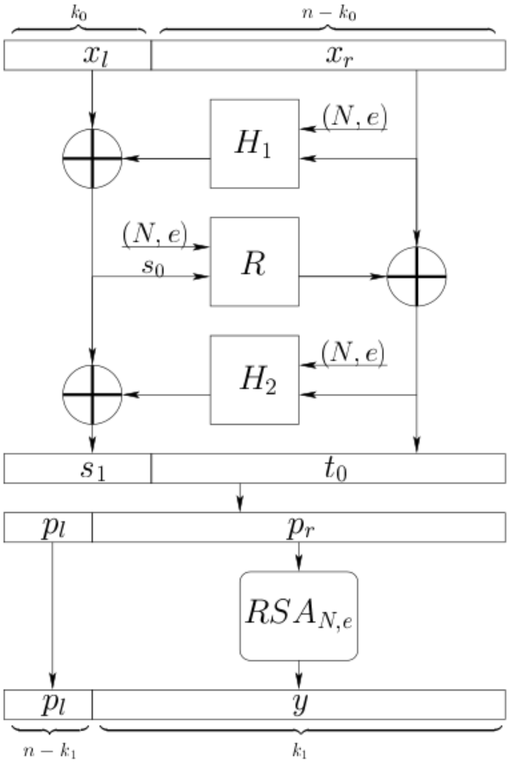
Where mpk is the max public-key probability of an encryption scheme AE. Thus, in short, EwH achieves PRIV-security if the starting encryption scheme is IND-CPA, and mpk_{AE}, or the likelihood that one can guess the keys of the scheme, is negligible.

RSA-DOAEP

The construction of RSA-DOAEP, where D is deterministic, is very similar to RSA-OAEP. A screenshot with the figure used in the paper is included at the end of this report. The major difference in the schemes are:

- Instead of including a random k_0-bit string, the parameters k_0 and k_1 are used in two stages.
- The input string is split into a k_0 length bit string, and a $n - k_0$ length bit string.
- Another round of hashing is used in the first phase.
- The outputs of the first phase are split into a k_1 length string, which is fed into the normal RSA function with N, and e as the encryption key. The remainder is simply forwarded to the result.

The security of RSA-DOAEP is proven starting with the statement: RSA-DOAEP achieves PRIV-security if RSA is one-way. The following proofs afterwards are based on that assumption. One noted benefit of RSA-DOAEP, as the authors state, is that RSA-DOAEP is length-preserving. The Encrypt-with-Hash scheme is not necessarily length-preserving, if the underlying encryption scheme is not length-preserving.



Efficiently Searchable Encryption, and Bucketization

In applying these two algorithms to the efficiently searchable encryption idea, the authors introduce two more concepts: perfect consistency, and computational soundness. The first is simply that the determinism produces the same results regardless of the conditions placed upon the algorithm. The second is that the probability of clashes in ciphertexts is at most $\delta(k)$, where $\delta(k)$ is a function based on the number of false positives possible in a given scheme. False positives, in this context, are cipher texts that can match to more than one plaintext message. The two deterministic schemes produced in this paper produce no false positives. However, the notion of PRIV as provided in the paper does not require the scheme to be deterministic, so such a measure is provided for schemes that are probabilitistic also.

With the allowance of deterministic algorithms allow indexes to be made on the ciphertexts, since the ciphertexts have no random element in it. However, this property is based on the plaintexts being encrypted have high min-entropy. A further step to adapt for low min-entropy plaintexts is a technique in database literature called *bucketization*, which in effect, is a reduction on the length of the hash, thus creating collisions in the ciphertext tags that are used as indexes. Intuitively, this simply means that due to collisions, an adversary cannot distinguish with high probability ciphertexts whose tags are equal.

CCA Extensions

The last section, the extension of the IND-CPA proofs to IND-CCA, mostly involve extensions of the provided IND-CPA schemes with additional caveats. One notable caveat being that no ciphertext can occur with too high a probability. Additional details of the CCA proofs are provided in this section, but with regards to the general gist of the paper, there are very few changes onto the original schemes provided.

Conclusion

To conclude this report, I have noted the main points in the paper that relate to encryption and database applications. The determinism in these public key schemes are shown to yield faster search times than probabilistic algorithms, due to the lack of possible collisions in the ciphertexts and their associated tags. I have not provided many of the details in the paper for purposes of time and length constraints, as those curious in the actual proofs can find them in the paper. To finalize this conclusion, the authors do their proofs in the random oracle (RO) model, and provide as an open question the construction of these proofs in the standard model.