1 General Properties of the Padding Scheme

- The expansion factor induced by a k by k Latin square is at most k.
- If the adversary does not know the Latin square applied, even in case he is given a set of pairs of plaintexts and ciphertexts, he cannot distinguish the remaining ciphertexts with success rate better than guessing.
- If he is not given any pairs of plaintexts and ciphertexts, any computationally unbounded adversary can only identify the plaintext-ciphertext pairs up to a set of permutations of size k. Furthermore, only one of the permutations gives the correct match between plaintexts and ciphertexts, all other permutations have no correct match.

2 Security Notions

There are two types of adversary in terms of power. The first adversary is curious-but-honest, who does not do anything out of the protocol. The second adversary knows some pairs of plaintexts and ciphertexts. In both case, the adversary tries to learn at least a pair of plaintext and ciphertext that he has not seen before. Both of the adversaries are defined in terms of games.

2.1 First Adversary

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^n)$
$d \leftarrow \mathcal{A}(1^n, \mathsf{pk})$
$ed \leftarrow \mathsf{Enc}(1^n, \mathsf{pk}, \mathsf{sk}, d)$
$(m, c) \leftarrow \mathcal{A}(1^n, \mathsf{pk}, d, ed)$
return $\mathsf{Enc}(m) = c$

In the game, the adversary generates the database as he wishes, and pass it to the challenger for an encryption. Given the encrypted database ed, the adversary's goal is to find a pair of plaintext and ciphertext. He wins the game if he can succeed with probability significantly greater than $\frac{1}{k}$.

Our padding scheme is secure against this adversary. The actual proof will consist of two parts, firstly the underlying ciphertexts are indistinguishable (in the usual sense), and secondly, the counts of the joint attributes do not reveal the plaintexts.

2.2 Second Adversary

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^n)$
$d \leftarrow \mathcal{A}(1^n, \mathsf{pk})$
$\{(m_i, c_i)\} \leftarrow \mathcal{A}(1^n, \mathsf{pk}, d)$
$ed \leftarrow \mathsf{Enc}(1^n, \mathsf{pk}, d)$
$(m, c) \leftarrow \mathcal{A}(1^n, \mathsf{pk}, d, ed, \{(m_i, c_i)\})$
return $\mathsf{Enc}(m) = c$

In this game, the adversary has the freedom to choose the database he wants to encrypt and a set of known plaintext-ciphertext pairs. His goal is to find a pair of plaintext and ciphertext he has not seen before. He wins if he can succeed with probability greater than one over size of unseen plaintexts, i.e. $\frac{1}{k-|\{(m_i,c_i)\}|}$.

Our padding scheme is not secure against this adversary if he can find Latin square we have applied, up to a permutation. By using a pair of plaintext and ciphertext, he is able to recover all plaintext-ciphertext pairs. In general, finding the Latin square used is NP-hard (any $k > 3$, and multiple attribute columns). Maybe we are giving the adversary too much power (knowledge and computation power)?

It is also provable that to be secure in this sense, one needs full padding.

3 Inference Attack

We need to define formally what is an inference attack (in general, not the technique). Then we want to show that given a secure scheme in our definition, there is no possible inference attack.

There are many ways to define inference attack. The most common attacks use accuracy of recovery as a measure.

4 Operations on the Database

We also want to understand what are the operations we can apply to the database and how they affect time and space complexities, and security.

5 Padding SSE Schemes on Relational Databases to Negate Inference Attack

$1:\quad (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_1(1^n)$
$2:\quad p \leftarrow \mathsf{KGen}_2(1^n)$
$3:\quad d' \leftarrow Pad(d, p)$
$4:\quad ed \leftarrow \mathsf{Enc}(1^n, \mathsf{pk}, d)$