**Ranking on Encrypted Data**

The scheme is to rank the keywords on their tf-idf weights. A ‘rank table’ indexing each keyword to its respective tf-idf in all the documents is created. All keywords in the rank table to be encrypted using any strong asymmetric encryption scheme. The rank table can be stored in a database in the untrusted cloud server.

Once rank table is created and keywords encrypted, all the tf-idf details in the table to be encrypted using ‘Onion Encryption scheme’, i.e., the plaintext tf-idf encrypted using ‘Order Preserving Encryption’, which in turn is then encrypted using ‘Paillier Encryption scheme’.

The reasons to go for this layered approach is detailed below,

1. ‘Order Preserving Encryption’ helps keep the order of values while also encrypting it at the same time. This will be a very useful property to help sort the encrypted values.
2. The second level ‘Paillier Encryption’ is critical for two reasons.
3. Firstly, by encrypting the first layer of encryption that is encrypted by OPE, it garbles the order that OPE maintained and thus even though order of the values is preserved in the first layer, with Paillier, the actual final encrypted values do not maintain any ordering and thus act as a first line of defense against any unauthorized intruder in the untrusted cloud server.
4. Secondly, the additive homomorphism of Paillier helps get the new values in case of multi-word keyword ranking, where we can add the encrypted values in ciphertext domain, decrypt it (to get the addition of OPE encrypted values) and then sort on them since they are OPE encrypted and maintain the order on addition, which can then be ranked.

So a typical search would go like this,

1. Client encrypts keywords using asymmetric encryption scheme and sends it to the server.
2. Server fetches the metadata corresponding to these encrypted keywords. Adds the values for multiple keywords for the same documents using Paillier encryption to get the final values. These values are decrypted and then ranked according to their magnitude and top-k document ids sent back to the client (or can directly send the documents itself)

**Limitation:**

Paillier keys need to be stored in server to decrypt the final values, which if accessed by an unauthorized entity can decrypt the top layer of encryption of the rank table. But, since there is one more layer of Order Preserving Encryption, he will only be able to reach the top layer of encryption and still will not be able to access the plaintext data in the rank table. However, once decrypted, the first layer of OPE may leak some information.

(Maybe we can think of a way to secure the Paillier keys, have to do more research)